PILOT MENTAL WORKLOAD CALIBRATION

THESIS

Jeremy B. Noel, Captain, USAF

AFIT/GOR/ENS/01M-12

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the U.S. Government.

AFIT/GOR/ENS/01M-12

PILOT MENTAL WORKLOAD CALIBRATION

THESIS

Presented to the Faculty

Department of Operational Sciences

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Operations Research

Jeremy B. Noel, B.S.

Captain, USAF

March 2001

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

AFIT/GOR/ENS/O1M-12

PILOT MENTAL WORKLOAD CALIBRATION

Jeremy B. Noel, B.S.
Captain, USAF

Approved:

Dr. Kenneth W. Bauer, Jr. (Advisor)

23 FEB 01
date

Jeffrey W. Lanning, Maj. (Reader)

23 Feb 01
date

Acknowledgements

This thesis effort represents over six months of hard work, and during that time I owe a debt of gratitude to many people. First, I would like to thank my thesis advisor, Dr. Kenneth W. Bauer, for introducing and solidifying my choice on this research topic through his genuine interest and observed energy in this research. Dr. Bauer continually challenged me to explore new ideas and I thank him for allowing me the latitude to focus this research in whatever direction I found most interesting. I would also like to thank my reader, Maj. Jeffrey Lanning, for his numerous corrections to improve the writing within this document. Dr. Wilson and Chris Russell have both been instrumental in helping me to understand the history and explore the future of the pilot mental workload problem. I thank them for their insights, assistance, and giving me the opportunity to spend time in the laboratory during simulated pilot mental workload experiments. In addition, I thank Joel Coons for providing dedicated computer support that, at times, went way beyond the call of duty.

Above all, however, I thank my wife for encouraging me to work hard throughout this whole thesis effort despite the recognition that every late night spent on this work was an additional night away from one another. In closing, I extend my thanks to family and friends for their support of my continued education and Air Force career.

Jeremy B. Noel

Table of Contents

Page

List of Figures

Figure Page

List of Tables

AFIT/GOR/ENS/01M-12

Abstract

The issue of predicting high pilot mental workload is important to the United States Air Force because lives and aircraft can be lost when errors are made during periods of mental overload and task saturation. Current research efforts use psychophysiological measures such as electroencephalography (EEG), cardiac, ocular, and respiration measures in an attempt to identify and predict mental workload levels. Existing classification methods successfully classify pilot mental workload using flight data from the same pilot on the same day but unsuccessfully classify workload using data from a different pilot on a different day.

The primary focus of this effort is the development of a calibration scheme that allows a small subset of salient psychophysiological features developed using actual flight data for one pilot on a given day to accurately classify pilot mental workload for a separate pilot on a different day. Extensive raw data preprocessing, including 29 Fourier transformations for each second of flight data, prepares the feature data for analysis. The signal-to-noise ratio feature screening method is employed to determine the usefulness of 151 psychophysiological features in feed-forward artificial neural networks. Factor analysis is used to identify patterns in features that vary with changes in mental workload level. Methodologies for workload level modification and data calibration are presented and tested to determine if any are useful in increasing the accuracy of measuring pilot mental workload across different pilots and over different days.

Through exploratory factor analysis, the reevaluation of the dimensions of the problem lead us to the insight that the feature space varies by pilot and day. While artificial neural networks appear unable to find this feature space by themselves, our calibration scheme exploits the new feature space and allows us to accurately discriminate between high and low mental workload. We achieve classification accuracy improvements over previous classifiers exceeding 55% while using 88% fewer features and reducing the classification accuracy variance by over 88%. Without the need for EEG data, the calibration scheme also reduces the raw data collection requirements by 99.75%, making data collection immensely easier to manage and dramatically reduces computational processing requirements. Along with the validated implementation method, the calibration scheme completely dominates all other classifiers over their entire operating curves and significantly simplifies the entire classification process. This makes the calibration scheme and implementation method far more practical than any previous classifier and classification method. Finally, the identification of the new feature space also opens new doors for further improvements in classification accuracies.

The calibration scheme produces a single classifier developed from only one flight that can be used to accurately predict pilot mental workload for different pilots over different days. The psychophysiological variations within and across individuals preventing previous methods from attaining high classification accuracy appear to no longer be a major hurdle.

PILOT MENTAL WORKLOAD CALIBRATION

I. Introduction

1.1 Overview

This research contributes to the advancement of knowledge regarding the problem of classifying pilot mental workload through the use of artificial neural networks. The goal of this research is to develop a calibration scheme that allows a parsimonious subset of salient psychophysiological features developed using data from a specific day to accurately classify pilot mental workload on a different day. In this context, parsimony means using the least number of features and saliency means selecting those features that have the strongest predictive power for classifying mental workload. A secondary goal is the development of a computer software tool that enables anyone using a standard office computer to perform the extensive preprocessing of the psychophysiological data quickly, accurately, and with minimum external software requirements. One proposed research question is: Can we develop a mental workload classifier that accounts for the psychophysiological differences across days with a single pilot? A second and more intriguing question is: Can we develop a mental workload classifier that is robust enough to account for the psychophysiological differences across days for multiple pilots?

This research effort uses data from a study conducted on several pilots flying identical aircraft over identical flight paths on two days, as well as several processes and methods developed in previous research work [10, 12, 15]. A saliency screening method will be employed on the psychophysiological features derived from this data to determine a parsimonious set of features for each pilot on each day [5]. Mental workload classification accuracies will then be measured following the training of artificial neural networks on these salient feature sets. Several methodologies for modifying the training and workload levels will be addressed, and a data calibration scheme will be presented. Finally, the different methodologies and data calibration scheme will be tested to determine if any are useful in increasing the accuracy of measuring pilot mental workload.

1.2 Background

With technological advancement in today's aircraft come increased demands on the pilots, often requiring their attention to be split between multiple tasks. When divided attention is coupled with stressful or mentally demanding situations, a potential for mental overload presents itself. Studies of fighter aircraft pilots show how devastating the effects of mental overload can be. These pilots can become so involved in their current situation that they forget to perform basic tasks, such as G-force straining maneuvers. As a result, some pilots have lost consciousness and their lives. One fighter pilot became so concerned about this problem that he conducted a study himself after surviving a G induced loss of consciousness (GLOC) incident [2]. He discovered that the

USAF lost fourteen pilots due to GLOC over ten years, with only one common factor found across the pilots: all but one of the fatalities occurred during mentally demanding portions of flight. If a classifier could be constructed to accurately analyze the psychophysiological data of the pilot and provide insight into the current level of mental workload, then a system could be developed to reduce the possibility of a GLOC situation.

The Air Force Research Laboratory (AFRL)/Human Effectiveness Directorate (HE) at Wright-Patterson Air Force Base, Ohio, has conducted many studies on mental workload in laboratory, simulator, and flight settings. Their results, used by the predecessors to this research effort, have indicated that the most influential psychophysiological features in classifying mental workload level are: brain electrical activity, heart rate, breath rate, and eye blink measures [28-32]. The AFRL has collected flight data using ten pilots flying Wright-Patterson Aero Club Piper Cubs on a specified route over two days. To collect the psychophysiological data, the pilots wore special recording equipment. Previous analysis of this data has revealed that substantial feature reduction is attainable through a signal-to-noise ratio feature-screening algorithm and that artificial neural networks produced the most robust classifier for determining mental workload [10, 15, 16]. While training an artificial neural network using these reduced features sets produced same-data mental workload classification accuracies varying from approximately 72%-97%, the classification accuracy for an individual pilot over multiple days using a classifier constructed from first-day data produced results comparable to flipping a coin [10].

1.3 Research Objectives

AFRL/HE has already performed an experiment and collected data on several pilots over two days. This psychophysiological data consists of electrical brain activity, heart rate, breath rate, and eye blink measures. Using feature selection techniques, artificial neural networks were trained with the hopes of accurately classifying mental workload. The resulting classification accuracies from classifiers built on data from one day used to predict mental workload with data from a second day were much lower than the desired 95% accuracy. This research concentrates on trying to solve this problem by developing a calibration scheme that can account for the psychophysiological differences pilots experience across days and therefore greatly increase the mental workload classification accuracy for trained artificial neural networks. This calibration scheme will then be used to evaluate the classification accuracy of multiple pilots over multiple days.

1.4 Research Methodology

While the specific methodologies of this research effort are included in Chapters III and IV, a quick overview of the approach is as follows:

- Preprocess the raw data into data files for each pilot on each day using only macros from Microsoft Excel and Word.
- Use artificial neural networks and the signal-to-noise ratio screening method to determine the most salient features from each data set, including mixed day data sets. With these networks, calculate performance measures across days and pilots.
- Investigate causes of low classification accuracy, such as challenging several assumptions concerning the threshold level between low and high workload, and develop a calibration scheme to overcome these difficulties.

- Validate the calibration scheme by calculating network performance measures on independent data and comparing the results to networks trained with non-calibrated data.

1.5 Scope of Research

As previously stated, the primary goal of this research effort is to develop a calibration scheme to allow a parsimonious subset of the most salient psychophysiological features developed from data on one day to accurately classify pilot mental workload on a different day. Additionally, this research effort provides the following:

- Development of a series of macros in Microsoft Office to perform the extensive preprocessing of the raw data
- Development of a process to identify and extract the middle layer node weights from Statistical Neural Network Analysis Package Version 2.0, the artificial neural network software tool
- Creation of an archive of all processed psychophysiological data, software tool files and instructions, and middle layer weight extraction process files.

A review of the literature concerning artificial neural networks, feature selection techniques, and the various psychophysiological features used in this research is addressed in Chapter II. Detailed information about the flight experiment and the extensive preprocessing requirements of the psychophysiological features is then presented in Chapter III. Chapter IV discusses the different methodologies followed to solve the classification accuracy problem, and the results are provided in Chapter V. The significance of the results, along with several conclusions and recommendations are then presented in Chapter VI.

II. Literature Review

This chapter reviews the pertinent literature involved in this research effort in four sections. The first section introduces artificial neural networks. The second section describes the feed-forward multilayer perceptron artificial neural network, followed by the third section that describes saliency screening methods for input features. Finally, the fourth section reviews the various psychophysiological features that are available when assessing mental workload in a multi-task environment.

2.1 Overview and History of Artificial Neural Networks

Artificial neural networks (ANNs) are inspired by how scientists believe brains function and organisms learn. It is well understood that the brain is composed of a network of interconnected neurons. Neurons receive simultaneous inputs from other neurons through their dendrites, causing some neurons to "fire" as they pass or suppress signals along the network [23]. The firing of various neurons, along with a changing network structure and weighting of the respective neurons, forms the basis for how organisms learn. This same concept of a network, including neurons connected to each other and interacting with one another simultaneously, is the structure and learning principle used in ANNs. Learning is accomplished by providing feedback to the network under supervised training to adjust the model parameters in order to provide more accurate model output.

Early users of ANNs, such as McCulloch and Pitts in 1943, created simple networks that involved neurons firing only when summed inputs exceeded bias threshold values [7]. In the 1950's, Rosenblatt challenged the models made by McCulloch and Pitts because they were single layer in nature, didn't take into account randomness inherent in many systems, and therefore only had limited capabilities and uses [7]. His ideas led to the development of Rosenblatt's perceptron, shown in Figure 2-1.

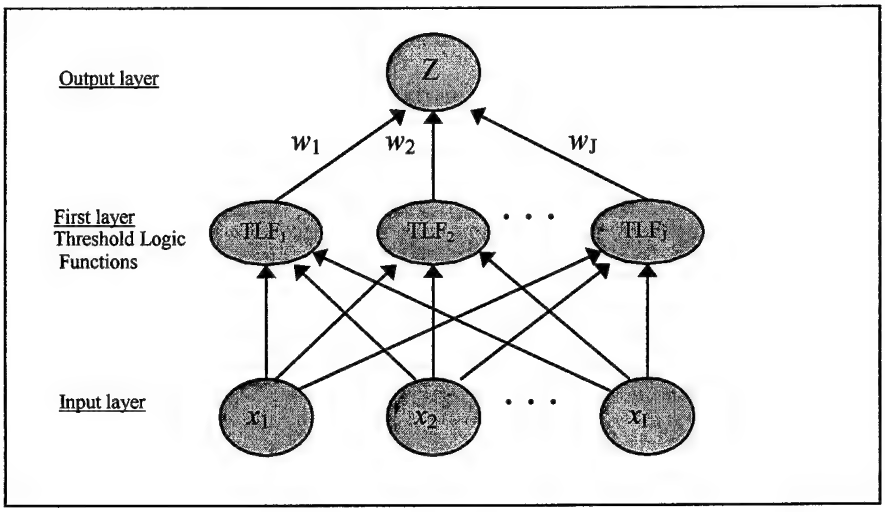


Figure 2-1. Rosenblatt's Perceptron

Rosenblatt's perceptron creates essentially a two-layer network (the input layer is not counted). The first layer contains fixed threshold logic functions, and the second layer provides the network output and has connecting trainable weights. Rosenblatt's perceptron improved an ANN's ability to distinguish between linearly separable functions, thus allowing it to perform adequately as a simple classification system. It still fell short, however, of accurately classifying regions that were not linearly separable, such as Exclusive OR (XOR) classification problems shown in Figure 2-2. In this case, the existing learning algorithms will never terminate, and any arbitrary stopping rules do not guarantee that the resulting weight vector from the network will generalize well for new data [7].

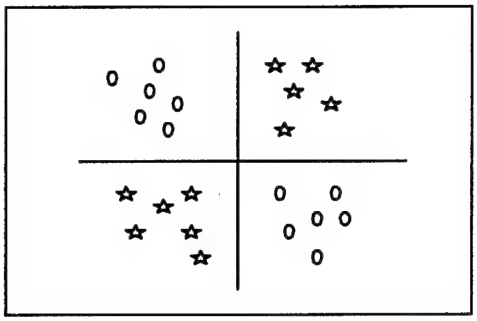


Figure 2-2. XOR Classification Problem

Minsky and Papert pointed out in 1969 that the reason these perceptron networks failed to correctly classify data sets that are linearly inseparable is due to the network structure only having a single layer of weights that are modified by the learning algorithm [7]. They showed that a network could solve a multi-dimensional problem, such as the XOR problem, as long as the number of perceptrons increased exponentially with the dimensionality of the problem being presented to the network. This would allow the ANN to operate in a transformed space where the problem can once again become linearly separable. Despite this discovery, size and computational limitations lead most researchers to believe that ANNs had little practical use for everyday problems and little progress was made toward improved learning algorithms or network structures until the late 1980's.

In 1986, Rumelhart, Hinton, and Williams announced the discovery of a new learning algorithm that eliminated the need for an exponential number of perceptrons to solve nonlinearly separable problems. Their approach, now called backpropagation, revitalized the ANN community by employing a gradient search method on the error surface produced following training. The gradient search method is implemented to minimize the error so that the network correctly classifies patterns as often as possible. Other modifications to backpropagation have been introduced since the late 1980's, but the backpropagation method has remained the most widely used algorithm by researchers and practitioners alike. As a result, the learning algorithm used in this research effort will also employ the backpropagation method.

With the renewed interest in ANNs and the development of increasingly more powerful computers, neural networks have been successfully used for complex pattern recognition. One particularly successful application includes recognizing patterns in psychophysiological data.

2.1.1 Definitions. Some basic definitions of terms used throughout this research effort are included below.

- Activation function. A mathematical function that takes the weighted activation values entering a unit, sums them, and translates the result to a position along a given scale [22]. Activation functions are generally chosen to be monotonic [7].
- Artificial Neural Network (ANN). An information processing system that operates on inputs to extract information and produces outputs corresponding to the extracted information [4].
- Architecture. The topological arrangement of neurons, layers, and connections, which defines the set of modeling equations available to the ANN [4].

- Backpropagation. A learning algorithm for a multiplayer perceptron (MLP) using gradient descent applied to the sum-of-squares error function, and updates the various network weights accordingly [7].

- Epoch. A complete presentation of the data set being used to train the MLP, also called a training cycle [22].

- Feature. Features refer to the input vectors of information that are presumed to have some relation for helping to distinguish the various output classes. A vector of features is often called an exemplar [4, 7].

- Feed-forward neural network. Multilayer ANNs whose connections exclusively feed inputs from lower to higher levels. In contrast to a feedback or recurrent ANN, a feed-forward ANN operates only until all the inputs propagate to the output layer, thus having the property that the outputs can be expressed as deterministic functions of the inputs. An example of a feed-forward ANN is the MLP [4, 7].

- Hidden unit. The processing element in MLP ANNs that is not included in the input or output layers. This part is located between the input and output layers [4].

- Learning algorithm. The algorithm that is used to train the ANN, resulting in changes to the weights of the neurons [7].

- Learning rate. A value established by the operator of the ANN that identifies how much the various weights can be changed after each training epoch in trying to minimize the squared error [7].

- Momentum. By adding the momentum term to the gradient search algorithm on the error surface, inertia is essentially added to the motion through the weight space. This "memory" of previous weight changes helps to avoid stopping at local minima on the error surface [7].

- Neuron. The fundamental building block of an ANN. Normally, each neuron takes a weighted sum of its inputs to determine its net input. The net input is then processed through its transfer or activation function to produce a single-valued output that is broadcast to neurons further down in the network [4].

- Sigmoid activation function. An activation function that squashes its input into a range usually set from 0 to 1 (thus allowing for network outputs to represent posterior probabilities when assuming the class-conditional densities can be approximated by normal distributions) [7].

- Weight. An indication of the strength or importance of a particular connection between neurons. Each processing element receives inputs by means of its connections, and each of these connections has an associated weight that identifies its strength [4, 7].

2.2 Description of a Feed-forward Multilayer Perceptron ANN

This research effort focuses on using feed-forward multilayer perceptron (MLP) ANNs, which consist of three layers: input layer, hidden layer, and output layer. Within a MLP ANN, a perceptron receives a weighted sum of I features and a bias term. The perceptron then transforms the weighted sum according to its activation function, producing the perceptron's output. The basic structure of this type of network, including the bias term, is shown in Figure 2-3.

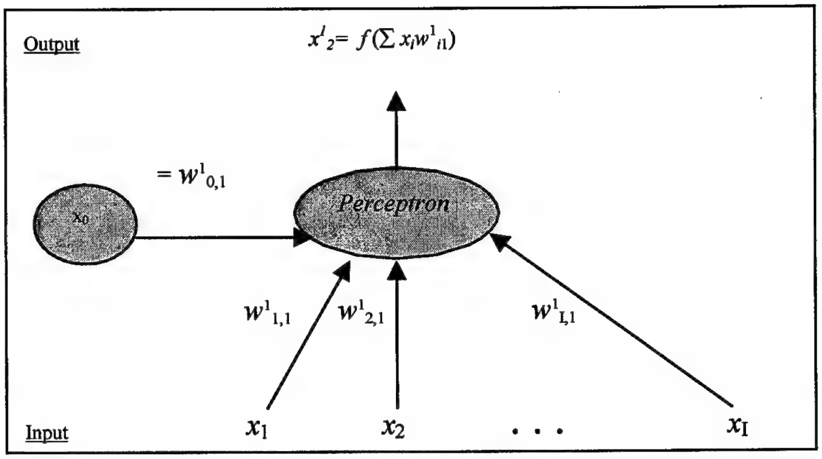


Figure 2-3. Single Perceptron with Bias Term.

As Figure 2-3 shows, data is fed upward from the input nodes x_1 through x_I towards the network output node. The network gets its name "feed-forward" due to the data always flowing forward through the network. The output y of the perceptron is found by executing the activation function after summing each neuron x_i for $i = 1, \ldots, I$ multiplied by its synaptic weight w_i for $i = 1, \ldots, I$ and adding the synaptic weight associated with the bias term w_0. The inclusion of the bias term allows the intercept to be non-zero. This equation is shown below.

$$\text{Output} = f[(\sum_{i=1}^{I} x_i * w_i) + \quad] \tag{2-1}$$

A more detailed explanation of some of the various components and important considerations in building and training MLP ANNs follows. In particular, network architecture, weight initialization and activation functions, and the backpropagation algorithm will be addressed.

2.2.1 MLP Network Architecture. The number of input nodes, hidden layers, hidden nodes, and output nodes define the architecture of a MLP ANN. Neural networks can be built with different architectures to solve the same problem, although some architectures are more effective in solving certain problems than others.

By convention, the number of features determines the number of input nodes for a network. Similarly, the number of classes the ANN is trying to classify determines the number of output nodes. The number of hidden layers used in an ANN can vary from none to many, however Bishop shows that a network with a single hidden layer is sufficient when approximating any multivariate problem [7]. Figure 2-4 shows a representation of a multivariate MLP ANN with a single hidden layer and a bias term.

The determination for the number of hidden nodes to use in a network, however, is not as clearly defined.

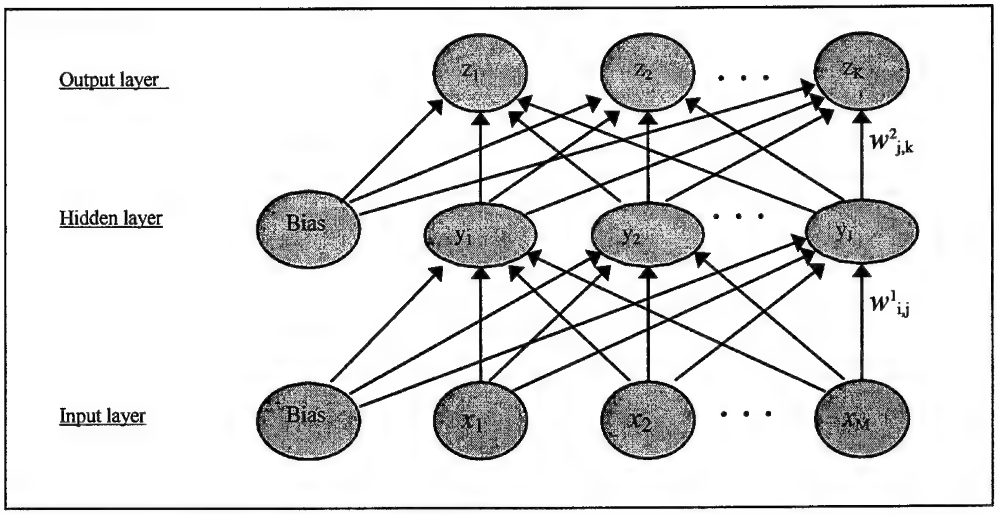


Figure 2-4. Multivariate MLP ANN with Bias Term.

Several algorithms and theories have been developed for selecting the "correct" number of hidden nodes for a particular network. The primary concern is that should a network be built with too few hidden nodes, then solution convergence is possibly compromised, and if too many hidden nodes are included, then the ability of the network to characterize new data might be reduced. As a general rule, Bishop argues that a network built with the number of hidden nodes equal to twice the dimensionality of the input space will result in an efficient network that is able to approximate any smooth mapping surface [7]. One algorithm for determining the upperbound for the number of hidden nodes is Kolmogorov's theorem. This theorem identifies that the number of hidden nodes needed for a network will never be more than twice the number of input nodes [7]. While other heuristic techniques have also been developed, the final determination of the "correct" number of hidden nodes to include in a network for a particular problem still remains somewhat more of an art form than a deterministic mathematical expression.

Besides the number of nodes and layers to include in a MLP ANN, issues such as raw feature data transformation, learning rate step-size, momentum rate values, weight initializations, and network training must also be considered. These issues will all be addressed in the remainder of this chapter.

With an understanding of the basic architecture used in building a MLP ANN, describing the general equation for calculating the output of the MLP ANN when presented with the *n*th input vector naturally follows.

The output from a MLP ANN for the *n*th input vector (z^n) can be computed by:

$$k\text{th neural network output} = z^n{}_k = f\left(\sum_{j=1}^{J} w^2_{j,k} * x^1_j\right) \tag{2-2}$$

where

- J is the number of hidden nodes.
- $f(a) = 1/(1+e^{-a})$ for sigmoidal activation functions, or $f(a) = a$ for linear activation functions.
- $w^2_{j,k}$ is the weight from the hidden node j to the output node k.
- x^1_0 is the hidden layer bias term and is set equal to 1.
- $x^1_j = f(\sum w^1_{i,j} * x^n_i)$ is the output of hidden node j and is summed from i = 1 to M.
- M is the number of input features.

- $w^{1}_{i,j}$ is the weight from input node *i* to hidden node *j*.
- x^{n}_{0} is the input layer bias term and is set equal to 1.
- x^{n}_{i} is the *i*th input feature of the *n*th input vector.

2.2.2 MLP Weight Initialization and Activation Functions. Before a MLP ANN can be used, the values of the weights between the input layer and the hidden layer, and between the hidden layer and the output layer must be assigned. This initial assignment is the only time the weights are dealt with directly. Afterwards, the backpropagation algorithm performs all modifications to the weights.

Smith found that randomly initializing the weights close to zero resulted in quicker training times for the ANN [19]. The case for the random assignment of the weights is due to the error calculations and subsequent weight modifications in the backpropagation algorithm. Briefly, if all of the weights in the network are initialized to the same value, then the hidden nodes all receive the same input values, the activation function calculations in the hidden layer all result in the same output values leading into the output layer, and the output layer values will all be identical. When the backpropagation algorithm calculates the partial derivative of the network output error with respect to the weight parameters, the network weights will all be updated identically, leading to an inability of the network to solve a nonlinear problem. Greene found that randomly initializing the weights between –0.05 and 0.05 worked best when the Signal-to-Noise Ratio (SNR) feature screening method was employed [12].

In order for the network to treat all inputs equally, the activation function must limit the inputs into a small range, usually –1 to 1 or 0 to 1. These modified inputs are then transmitted from the hidden node layer to the output layer through the weighted branches. Examples of activation functions include: hard limiter, threshold logic, hyperbolic tangent, and sigmoid. Graphs of each of these activation functions are shown in Figures 2-5 through 2-8.

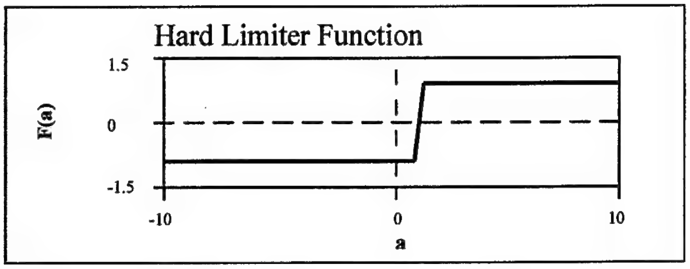


Figure 2-5. Hard Limiter Activation Function.

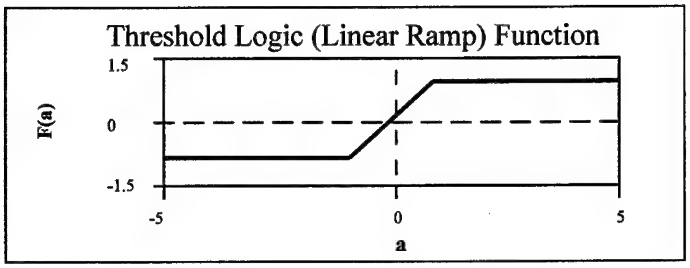


Figure 2-6. Threshold Logic (Linear Ramp) Activation Function.

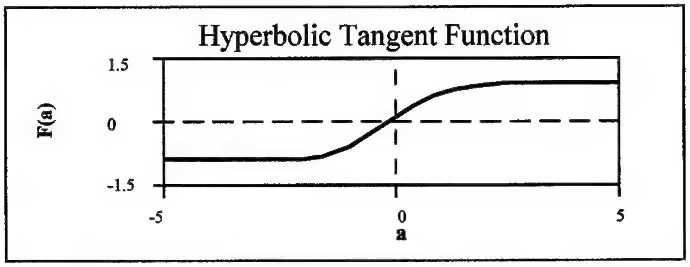


Figure 2-7. Hyperbolic Tangent Activation Function.

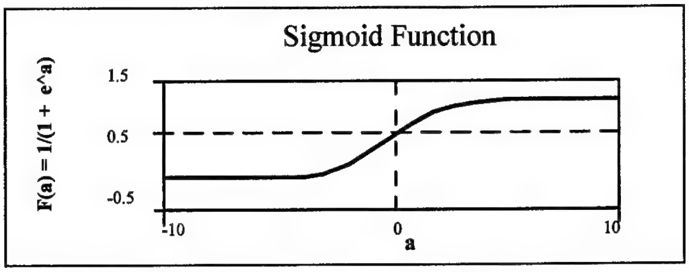


Figure 2-8. Sigmoid Activation Function.

Notice how the hard limiter and threshold logic functions are linear in nature, while the hyperbolic tangent and sigmoid functions are non-linear. This non-linearity allows for a continually differentiable function and is therefore more desirable. For the purposes of this research effort, the sigmoid activation function is used exclusively due to its robust nature.

2.3 *The Backpropagation Algorithm.*

In order for an ANN to be useful in classifying exemplars, the network must first be trained. The one most widely studied training algorithm and also the one used exclusively in this research effort is called backpropagation [7]. Training any neural network involves an iterative process by which the network receives inputs, pumps them through the network using the current weight values, calculates the network outputs and the resulting error values based on comparisons with the known outputs, and then modifies the various nodal weights throughout the network in efforts of reducing the calculated error. The backpropagation method is simply one algorithm by which the weights are updated throughout the network.

The cornerstone of the backpropagation algorithm lies in differentiable activation functions, such as the sigmoid activation function used for this research effort. This is important because the activations of the output nodes become differentiable functions of both the input variables, and of the weights and biases [7]. If we apply an error function, such as a sum-of-squares error function, a differentiable function of the network output is created and the error is a differentiable function of the weights [7]. By evaluating the derivatives of the error function with respect to the different weights, we then find weight values that minimize the error. The algorithm that evaluates these derivatives and updates the various weights is the backpropagation algorithm, and it uses a gradient descent approach to find the minimum error on the error surface. The actual updating of the weights can occur in two ways: an instantaneous update that examines the gradient of the error surface after the network processes each training exemplar, and a batch method that examines the gradient of the error surface only after the network has processed all of the training exemplars [7]. The method used in this research effort incorporates the instantaneous update method, and an algorithm using this method is provided below [20].

1. Randomly partition data into training, training-test, and validation data sets.
2. Normalize the feature input data.
3. Initialize the weights to small random values.
4. Present the network with a randomly selected exemplar from training set, denoted $\boldsymbol{x}^P$.
5. Calculate the network output, $\boldsymbol{z}^P$, associated with the pth training vector.
6. Update the weights.
7. If the training-test data set error does not indicate sufficient convergence, go to step 4.

The first step of this algorithm involves randomly partitioning the entire data set into three separate data sets: training, training-test, and validation. The training set consists of the data that will be presented to the ANN for updating the weights, and a portion of this data will be held back for assessing network performance. This hold out data is called the training-test data set. The validation data set is used to independently measure how well the ANN predicts future responses and produces the expected outputs.

The purpose of the training-test data set is to identify when the network is overfitting the data. Overfitting means that the ANN is becoming so finely tuned to the training data set that it is "memorizing" even the noise in the data set. This is not necessarily a problem to the user, depending on the purpose of building the ANN. If the purpose is to build an ANN that can very accurately classify an exclusive set of data, then overfitting this particular data set might be warranted. Under this circumstance, the performance of the ANN would be excellent for the training and training-test data sets, while very poor for the validation data set. Overfitting the data can be a concern, however, if the intent is to build a robust ANN that can accurately classify data outside of the training data set.

There are many different ways to divide the whole data set into the three data sets described above. One method involves splitting the data set into the training and validation sets using a 2:1 ratio. If one then splits the training set again by a 2:1 ratio, the creation of the training and training-test sets will have been accomplished, and a 2:1:1.5 ratio will result across the training, training-test, and validation data sets. Another method results in a 40/30/30 split across the training, training-test, and validation sets, respectively. The way one decides how to best split the data depends on the number of available exemplars and the particular application of the ANN. Larger data sets allow for the use of the 2:1:1 ratio, while smaller data sets might only allow for a 40/30/30 or similar split.

With the data split into the different sets, the second step of the algorithm normalizes the feature input data. Two basic approaches can be taken to accomplish this step: scaling the data to fall within a range (like –1.0 to 1.0, or 0.0 to 1.0), or standardizing each feature to a mean of 0.0 and a variance of 1.0 [27]. Steppe recommends normalizing the data sets independently, which will keep the test and validation sets as separate and independent of one another as possible [20].

The third step of the algorithm involves initializing the weights within the ANN to small random values. The purpose behind the randomness, and a suggested range of values for the weights has already been addressed in an earlier section. In the fourth step, a randomly selected exemplar from the training set, denoted x^P, is presented to the network. This exemplar is the *p*th vector from this set. During the fifth step of the algorithm, the network calculates the output from this exemplar, denoted z^P, which is the output associated with the *p*th training exemplar. Equation 2-2 detailed this output function as a summation of the sigmoid activation functions and the current weights in the network. The sixth step in the algorithm updates the weights in the network. Section 2.3.1 details how the network updates the weights.

The seventh step in the backpropagation algorithm tests to see if the weights have converged sufficiently to stop the training network. The training-test data set is used for this test. If the average error distance (the difference between the observed output and the actual output) for the most recent interval is less than the average error distance over a previous fixed interval, then training should continue by repeating steps four through seven. If the average error distance for the most recent fixed interval is not less than the average error distance over a previous fixed interval, then training should be stopped. Any continued network training from this point onward is unlikely to produce better results due to an overtrained network, and the weights should be left with the values that produced the minimum error on the training-test sample [19]. Other methods to stop a network from training include reaching a maximum number of training epochs and the attainment of training error target value [27].

2.3.1. Updating Weights in the Backpropagation Algorithm. This section details how the weights are updated in the backpropagation algorithm once the network output, $\mathbf{z}^{P}$, associated with the pth training exemplar is calculated.

The weight updating is accomplished by calculating the instantaneous output error, ε^{p}_{o}, associated with $\boldsymbol{x}^{p}$ from the *p*th exemplar of observed outputs, z^{p}_{k}, and the corresponding vector of desired outputs, $\boldsymbol{d}^{p}_{k}$. In this case, p represents the *p*th input exemplar of data, and k represents the number of output nodes, which is typically equal to the number of classes one is trying to classify. The formula to calculate the instantaneous network output error, ε^{p}_{o}, is the square error associated with the *p*th exemplar, shown below:

$$\varepsilon^{p}_{o} = \sum_{k=1}^{k} (d^{p}_{k} - z^{p}_{k})^{2} \qquad (2\text{-}3)$$

where

- K is the number of output nodes
- d^{p}_{k} is the desired output vector associated with the *p*th input exemplar
- z^{p}_{k} is the observed output vector produced from the *p*th input exemplar at the *k*th output

Using this defined error surface, the gradient descent step direction is found by taking the partial derivative of the error surface with respect to the weights currently in the network. There are four different calculations for the partial derivatives of the error surface, δ, depending on the layer of weights being updated and the type of activation function used. Equations 2-4 and 2-5 reflect δ when using sigmoidal activation functions, and equations 2-6 and 2-7 reflect δ when using linear activation functions.

Equation for weights between input and hidden layers (sigmoid function) are

$$\delta^{1}{}_{\mathrm{k}} = x^{1}{}_{j}(1\text{-}x^{1}{}_{j})\Sigma\delta^{2}{}_{\mathrm{k}}(w^{2}{}_{jk})^{\mathrm{old}} \quad \text{for k = 1,...K} \tag{2-4}$$

where $(w^{2}{}_{jk})^{\mathrm{old}}$ is the old weight from hidden node j to output node k.

Equation for weights between hidden and output layers (sigmoid function) are

$$\delta^{2}{}_{\mathrm{k}} = (d^{p}{}_{k} - z^{p}{}_{k})z^{p}{}_{k}(1\text{-}z^{p}{}_{k}) \tag{2-5}$$

Equation for weights between input and hidden layers (linear function) are

$$\delta^{1}{}_{\mathrm{k}} = \Sigma\delta^{2}{}_{\mathrm{k}}(w^{2}{}_{jk})^{\mathrm{old}} \quad \text{for k = 1,...K} \tag{2-6}$$

Equation for weights between hidden layer and output layers (linear function) are

$$\delta^{2}{}_{\mathrm{k}} = (d^{p}{}_{k} - z^{p}{}_{k}) \tag{2-7}$$

Using this gradient descent direction, the weight parameters in the network can then be updated. Once again, there are two equations to update the weights, depending on the location of the weights. Equations 2-8 and 2-9 identify the weight updating equations between the input and hidden layers, and the hidden to output layers.

Weight update equation for weights between the input and hidden layers:

$$(w^1{}_{ij})^{\text{new}} = (w^1{}_{ij})^{\text{old}} + \eta\delta^1{}_j \mathrm{x}^p{}_i \tag{2-8}$$

Weight update equation for weights between the hidden and output layers:

$$(w^2{}_{jk})^{\text{new}} = (w^2{}_{jk})^{\text{old}} + \eta\delta^2{}_k x^1{}_j \tag{2-9}$$

where

- $(w^1{}_{ij})^{\text{new}}$ is the updated weight from input node i to hidden node j.
- $(w^1{}_{ij})^{\text{old}}$ is the old weight from input node i to hidden node j.
- $(w^2{}_{jk})^{\text{new}}$ is the updated weight from hidden node j to output node k.
- $(w^2{}_{jk})^{\text{old}}$ is the old weight from hidden node j to output node k.
- η is the learning rate, or the training step size.
- $x^1{}_j = f(\sum w^1{}_{ij}\, x^p{}_i)$ is the output of hidden node j (i = 1, …, M)
- $x^p{}_i$ is the i^th^ input feature of the *p*th input vector.

The learning rate, η (defined in Section 2.1.1), measures how quickly the ANN will try to reduce the error during each backpropagation cycle. It ranges from zero to one, indicating the proportion of error that will be reduced during each weight updating cycle. If a learning rate with a value close to zero is used, then small steps in the gradient search will be taken. This leads to long convergence and computational times, and is therefore rather inefficient. On the other hand, using a value close to one entails large steps in the gradient search, leading to possibly overshooting the minimum (which might actually cause an increase in the error). Furthermore, using the backpropagation algorithm with large training values can cause divergent oscillations and an inability for the network to stabilize at a solution [7]. A constant learning rate value set at $\eta = 0.25$ has shown to be relatively efficient in terms of computational time and convergence speed [12, 15].

The backpropagation algorithm can sometimes get stuck with solutions on the error surfaces that are local minimums instead of global minimums. To help avoid this problem, practitioners add a momentum term, α, into the backpropagation equations. The momentum term allows a network to respond to both the local gradient as well as recent trends in the error surface through the effect of inertia [7]. The term makes changes to the weights equal to the sum of a fraction of the last weight change and the new change suggested by the backpropagation rule [15]. Weight updating Equations 2-10 through 2-13 are modified to incorporate the momentum term.

Equation for weights between the input and hidden layers are

$$[w(t+1)^{1}{}_{ij}]^{\text{new}} = [w(t)^{1}{}_{ij}]^{\text{old}} + \eta \delta^{1}{}_{j}\, x^{p}{}_{i} + \alpha\, \Delta[w(t-1)^{1}{}_{ij}]^{\text{old, old}} \qquad (2\text{-}10)$$

Equation for weights between the hidden and output layers are

$$[w(t+1)^{2}{}_{jk}]^{\text{new}} = [w(t)^{2}{}_{jk}]^{\text{old}} + \eta \delta^{2}{}_{k}\, x^{1}{}_{j} + \alpha\, \Delta[w(t-1)^{2}{}_{jk}]^{\text{old, old}} \qquad (2\text{-}11)$$

$$\Delta[w(t-1)^{1}{}_{ij}]^{\text{old, old}} = [w(t)^{1}{}_{ij}{}^{\text{old}} - w(t\text{-}1)^{1}{}_{ij}{}^{\text{old, old}}] \qquad (2\text{-}12)$$

$$\Delta[w(t-1)^{2}{}_{jk}]^{\text{old, old}} = [w(t)^{2}{}_{jk}{}^{\text{old}} - w(t\text{-}1)^{2}{}_{jk}{}^{\text{old, old}}] \qquad (2\text{-}13)$$

where

- α is the momentum term.
- $[w(t+1)^{1}{}_{ij}]^{new}$ is the new weight at epoch (t+1) from input node i to hidden node j.
- $[w(t+1)^{2}{}_{jk}]^{new}$ is the new weight at epoch (t+1) from hidden node j to output node k.
- $[w(t)^{2}{}_{jk}]^{old}$ is the old weight at epoch t from hidden node j to output node k.
- $[w(t)^{1}{}_{ij}]^{old}$ is the old weight at epoch t from input node i to hidden node j.
- $\Delta[w(t-1)^{1}{}_{ij}]^{old,\ old}$ is the weight change from epoch (t-1) to epoch t for input node i to hidden node j.
- $\Delta[w(t-1)^{2}{}_{jk}]^{old,\ old}$ is the weight change from epoch (t-1) to epoch t for hidden node j to output node k.
- t is the training epoch

These equations show a momentum rate set to zero causes the weights to change exactly as they would when using only the error gradient. A momentum rate set to one, which is the highest value that should ever be used, will cause the weights to change the same amount from the previous update plus the current gradient step. Any values greater than one will result in an exponential impact on training [27]. Since the momentum term and learning rate together often determine the extent of the adjustments the weights will experience with each update, the settings for these values should be carefully considered in light of one another. The learning rate determines the magnitude of the next gradient step, while the momentum term determines how much the previous step will impact the next step. Other research efforts have shown that setting the momentum term around $\alpha =$ 0.9 results in quick training times when the learning rate is set around $\eta = 0.25$ [12,15].

Building a network with a larger momentum term implies that a smaller learning rate should be used [27].

2.4 Feature Selection and Reduction Using Saliency Measures.

In order for an ANN to produce good results, only the best features should be presented to the network for training. If only a few features are available for use in the training process, reducing them to an even smaller number is likely unnecessary. Presenting too many features to the network, however, can result in an ANN with poor classification accuracy. This is especially true if there is a large amount of noise in the data. As a result, several measures and methods have been developed to assist in determining which features are most salient. Three of these measures are Ruck's saliency measure, Tarr's saliency measure, and the signal-to-noise ratio (SNR) saliency measure.

Each saliency measure uses its own equations and algorithm. Ruck's measure is based on using the partial derivatives from a trained network output with respect to the feature inputs over a number of independently trained networks [15]. The result produces ranked features according to their average saliency metric over several training cycles [6, 21]. Tarr's measure is based on using the sum of the squared weights between the input and hidden nodes, and also produces ranked features according to their saliency metric [10]. The SNR measure is also based on the sum of the squared weights connecting the input and hidden nodes and compares the weights of each feature to the saliency of an injected noise feature [5]. While any of these feature saliency measures could be used, this research effort focuses on using the SNR saliency measure.

2.4.1 Signal-to-Noise Ratio Saliency Measure. As introduced above, the SNR saliency measure involves summing the squared weights connecting the input and hidden nodes, and then comparing the sum from each feature to the saliency of an injected noise feature. The metric's computation is shown in equation 2-14.

$$SNR_i = 10\log\frac{\sum_{j=1}^{J}(w_{ij}^1)^2}{\sum_{j=1}^{J}(w_{Nj}^1)^2} \qquad (2\text{-}14)$$

where

- SNR_i is the saliency metric for the i^{th} feature
- J is the number of hidden nodes
- $w^1{}_{Nj}$ is the weight connecting the injected noise feature, x_N, to the hidden node layer
- $w^1{}_{ij}$ is the weight connecting the input feature, x_i, to the hidden node layer

In order to use the SNR saliency measure, the noise feature must be added to the data set. The uniform (0,1) distribution is used for this purpose [5]. The concept behind why the SNR saliency measure works lies in the movement and size of weights as the backpropagation algorithm tries to reduce the error. Features that are relevant to the ANN's output will have weight values in the first layer that are significantly greater than features with little relevancy to the output, whose weight values should fluctuate around zero [5]. As a result, SNR values for salient features will be larger than SNR values for non-salient features, and ranking the features by their SNR values can be accomplished.

2.4.2 Signal-to-Noise Ratio Screening Method. A method using the SNR concept has been developed with the purpose of identifying a parsimonious set of salient features. To do this, non-salient features must be removed from the data set while still allowing the ANN to generalize the data set well. The method used to accomplish this feature reduction is shown below [5, 12].

1. Introduce a Uniform (0,1) noise feature, x_N, to the original set of features.
2. Standardize all features to zero mean and unit variance.
3. Randomly initialize the weights between –0.001 and 0.001.
4. Randomly select the training and test sets.
5. Begin to train the ANN.
6. After each epoch, compute the SNR saliency measure for each input feature.
7. Interrupt training when the SNR saliency measures for all input features have stabilized.
8. Compute the test set classification error.
9. Identify the feature with the lowest SNR saliency measure and remove it from further training.
10. Continue training the ANN.
11. Repeat steps 6-9 until all features (except the noise feature) in the original set are removed from training.
12. Compute the reaction of the test set classification error due to the removal of the individual features.
13. Retain the first feature whose removal caused a significant increase in the test set classification error, as well as all features that were removed after that first salient feature.
14. Retrain the ANN with only the parsimonious set of salient input features.

Using the SNR screening method described above allows for a quick screening of the input features at any time in the network training process. As a result, when one is presented with many features, this screening method quickly eliminates the non-salient features before excessive training time has been wasted. Other screening methods require multiple independently trained networks to eliminate features, where as the SNR screening method only needs one. In addition, several studies have found that the SNR screening method produces robust results [5, 12]. This indicates that another advantage to the SNR screening method is its robustness when compared to other more statistically rigorous screening methods.

2.5 Psychophysiological Features

United States and foreign industries, along with governmental agencies, have long been interested in the effects of mental workload on animals and humans [1, 2, 3, 10, 11, 12, 14, 15, 16, 25, 28, 29, 30, 31, 32]. To observe the effects of mental workload, many prominent psychophysiological features have been developed and studied. While other measures and features exist, the focus of this study revolves around using features derived from these four measures: cardiac, respiratory, ocular, and brain activity. Furthermore, research has shown that using multiple psychophysiological features simultaneously provides a more complete mental workload picture of a test subject in a multi-task situation, such as flying an airplane, than any single feature by itself [10, 11, 15, 16, 29, 31].

2.5.1 Cardiac Measures. Using the heart to measure physical and mental workload is not a new concept. In fact, studies dating back to the early 1930's have used the heart rate to assess pilot responses [28]. This is primarily due to the ease with which data can be gathered since cardiac measures can be taken non-intrusively and are continuously available [28]. As a general guideline, increases in heart rate have been associated with increases in mental workload. During flight, pilots may experience increased heart rates when performing more difficult or more demanding operations such as take offs and landings [14, 29, 31].

Another cardiac feature recorded for estimating mental workload is heart rate variability. Heart rate variability is simply the variation of the beat-to-beat heart rhythm. It is not a statistical calculation of heart beat variance, but rather a measure of how much the heart inter-beat intervals change. Generally, this beat-to-beat variability decreases with increased mental workload, and increases with decreased mental workload [28]. Despite the number of studies that have measured heart rate variability, some controversy remains regarding its practical use. The controversy extends from how to best calculate the measure to research conclusions that heart rate variability provides no additional information beyond what can be gleaned from heart rate alone [28].

2.5.2 Respiratory Measures. Studies using respiratory measures have found a general increase in respiration during periods of higher mental workload [28]. Despite the general connection between respiration and workload, however, respiratory measures have not been widely used to estimate cognitive workload. One reason involves the complexities associated with removing the effects of speech and physical activity on a test subject's breathing pattern [28]. As a result, increased respiration rates appear to be an indication of increased workload conditions, but the collection, processing, and interpretation of the data can be difficult.

2.5.3 Ocular Measures. The most common features using ocular measures include duration of eye blinks and eye blink rate. Past research has shown that as test subjects attempt to process more information due to high visual workload demand, their blink rate and blink duration decrease [14, 31]. In other words, as the visual demands increase in the environment, test subjects must focus their attention more to avoid missing important information. Furthermore, research has been published indicating that blink rate is possibly more sensitive to cognitive workload levels than blink duration [31]. Blink duration, on the other hand, appears to be more dependent on the amount of visual information presented to test subjects than blink rate [31]. Any variations in these features are most noticeable, however, when visual demands vary and are overall not as sensitive to auditory or cognitive workloads where less visual stimulation is involved [31].

2.5.4 Brain Activity Measures. In recent years, given the ever-increasing computational power of computers, researchers have been able to process and analyze data from a brain like never before. Through the use of electrodes, the electrical impulses spanning the brain can be recorded and electroencephalographs (EEGs) generated [30]. These graphs plot the voltage changes over time at a particular location of the brain [3]. With this information, researchers have successfully used EEG data to monitor workload in multi-task environments [10, 11, 16, 30, 31, 32]. The frequency range found to be most associated with cognitive workload lies from 1 to 40 Hertz (Hz), with frequencies below 1 Hz generally thought to be due to eye movements and frequencies over 40 Hz due to muscle movements. Furthermore, the cognitive workload frequency range can be broken down into 5 distinct power bands, shown in Table 2-1 below.

Table 2-1. EEG Frequency Power Band Designations.

Band	**Symbol**	**Frequency**
Delta	Δ	1 – 3 Hz
Theta	θ	4 – 7 Hz
Alpha	α	8 – 12 Hz
Beta	β	13 – 30 Hz
UltraBeta	$\mu\beta$	31 – 42 Hz

Using a Fourier transform, the raw EEG data can be transformed from a composite waveform into these 5 power bands. This is accomplished through a Fast Fourier Transform, which is a computationally efficient discrete Fourier transform algorithm [8]. The result is a conversion of the EEG data from a time-domain waveform to a frequency-domain waveform, upon which the 5 power bands are filtered for each second of recorded EEG data. Research using these power bands has shown that as cognitive demand increases, EEG activity in the alpha (α) band tends to decrease and EEG activity in the theta (θ) band tends to increase [3, 14, 32].

2.6 Chapter Summary

This chapter introduced the literature used as a foundation for this research effort. ANN architectures and learning algorithms are addressed, along with saliency screening methods for input features. The psychophysiological features required to classify mental workload are also presented. Chapter III discusses the flight experiment and the necessary data preprocessing that must be completed prior to training ANNs.

III. Data Collection and Preprocessing

This chapter includes information on the experiment and the data collected by the Flight Psychophysiology Laboratory (FPL) in the Human Effectiveness Directorate at AFRL. The methodology and software tool employed to preprocess the data is also discussed, and an example of a final input data matrix is presented. To ensure consistency, the preprocessing methodology is the same as used by both Laine and East [10,15].

3.1 The Flight Experiment

The data used in this analysis came from an experiment conducted by the AFRL/FPL on pilots at the Wright-Patterson Aero Club. Ten volunteers flew a predetermined flight route once a day for two days. Each flight, lasting approximately 44 minutes, was divided into 22 two-minute flight segments. Along with the pilot, a technician from the FPL and a copilot flew on each flight. The technician's job was to monitor the data collection process, and the copilot was present only for safety reasons and was not part of the experiment. While ten pilots participated in the flight experiment, only the data from Pilots 1 and 4 are analyzed during the course of this research effort.

The flight route was specifically chosen to include three levels of workload: low, medium, and high. The laboratory personnel graded the difficulty of each flight segment before the flight, and the test subjects graded the difficulty of the flight segments after the flight. Figure 3-1 shows a graph reflecting the pilot's subjective measures of workload level associated with each flight segment. Understandably, there were some discrepancies between the researchers and the pilots concerning workload levels associated with each flight segment. As an example, the pilots classified both the IFR airwork and VFR touch-and-go segments as high workload levels, while the researchers classified the VFR touch-and-go segment as high workload and the IFR airwork as medium workload.

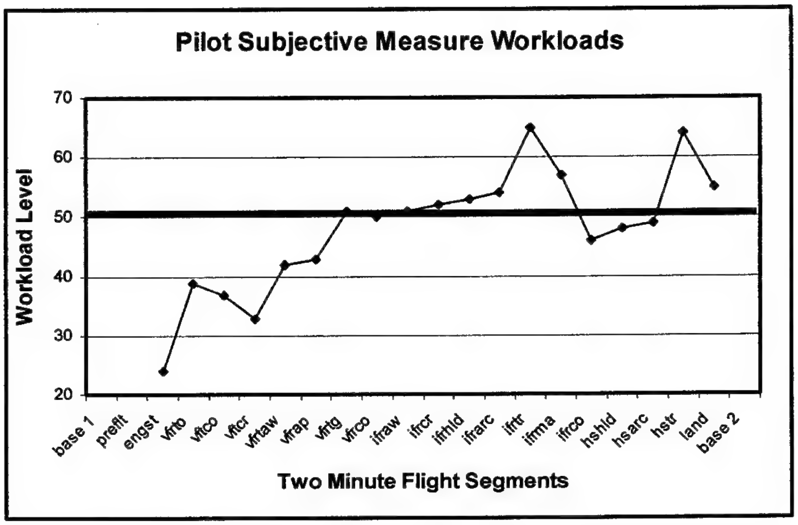


Figure 3-1. Pilot Subjective Measure Mental Workload Ratings

To rectify the difference, the pilots and researchers agreed that since both groups classified the touch-and-go segment of the flight as high workload, then this would be the minimum threshold for determining a high workload segment. East found classifying three workload levels (low, medium, and high) very difficult and combined the low and medium levels into one group called low workload [10]. As a result, the dark horizontal line drawn across Figure 3-1 separates the low and high workload levels. All of the flight segments below the line were classified as low mental workload and all of the flight segments above the line were classified as high mental workload.

With the creation of this line, however, two significant assumptions are made concerning workload level accuracy and transitions between flight segments that can significantly increase classification errors. The first assumption deals with how accurately the flight segments are classified by mental workload for the pilots. It is assumed that all flight segments classified as low mental workload are equal in workload to other low workload flight segments. Similarly, it is assumed that all flight segments classified as high mental workload are equal in workload to other high workload flight segments. Determining the true mental difficulty for individual flight segments is not a science, however, and it is possible that the compromise between the researchers and pilots results in inaccurate workload levels. Chapter IV explores different schemes for defining the workload states to identify the effects of this assumption.

The second assumption deals with instantaneous transitions between flight segments where the low/high workload line is crossed. It is assumed that the transition from low to high (or high to low) workload is instantaneous. In other words, the last second of the previous flight segment is correctly classified as low, and the first second of the following fight segment is correctly classified as high. However, transitions between mental workload levels are not really instantaneous since they occur over time and can vary by individual pilot. The effects of this assumption are also addressed through the different schemes for defining the workload states in Chapter IV.

3.2 Psychophysiological Data Collected

Four different types of psychophysiological data are collected during flight: EEG data, ocular data, respiratory data, and cardiac data. To collect the EEG data, the pilots wear a special cap on their heads fitted with 29 electrodes. Figure 3-2 shows a diagram of a pilot's head fitted with the electrodes. Each of these electrodes has an identifier associated with it that reflects the location and naming of the electrode site based on the International 10-20 system [15]. The letter of each identifier designates the brain region and the number provides location information relative to the left or right side of the brain. An even number identifies the electrode to be on the right side of the brain and an odd number means the electrode is on the left side. The larger the number, whether odd or even, means the electrode is further from the center of the brain, where the center runs from the nose to the back of the head. A "Z" designates a central location, and the middle of the brain has no numerical designator. Table 3-1 lists the different regions of the brain associated with the letters found in the electrode identifiers.

The ocular, respiratory, and cardiac data are recorded in data files that contain the elapsed time in milliseconds between events. An event is simply the blink of an eye, the taking of a breath, or a beat of the heart. A few additional pieces of raw data are also made available for several of these features, including the maximum and minimum amplitudes associated with each breath, and the amplitude and duration of each eye blink.

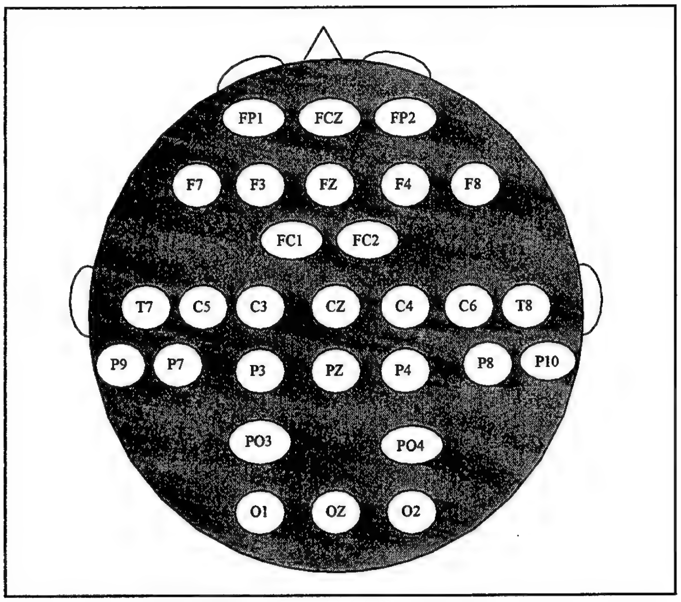


Figure 3-2. EEG Electrode Locations as Viewed from Top of Head

Table 3-1. Regions of EEG Identifiers

Letter	Location
C	Central
F	Frontal
O	Occipital
P	Parietal
T	Temporal

3.3 EEG Processing

The raw EEG data is collected and immediately sent through a program called *Manscan* 4.0, which filters out some of the undesirable artifacts from the EEG signals. Examples of these artifacts include muscle movements such as movement of the pilot's head moving during flight and eye movement. At this point, the EEG data is saved into large data files, one file for each of the 22 flight segments, for more thorough processing. This processing, described in greater detail below, ignores two extraneous data columns also stored in the data files: the Horizontal Electro-oculography (HEOG) and the Vertical Electro-oculography (VEOG). These two columns record the movements, both horizontally and vertically, of the pilot's eye during flight and are not considered indicators of mental workload. Instead, the *Manscan* program uses these columns of data to remove the undesirable artifacts due to eye movement, and consequently, they can be deleted or ignored during the remaining preprocessing of the EEG data. An example of the raw EEG signal data for one node over a 0.5 second interval is shown in Figure 3-3.

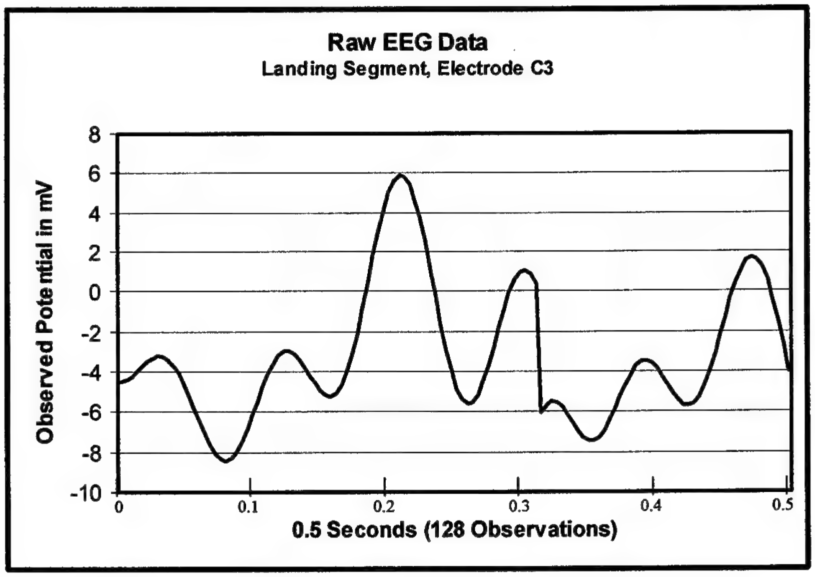
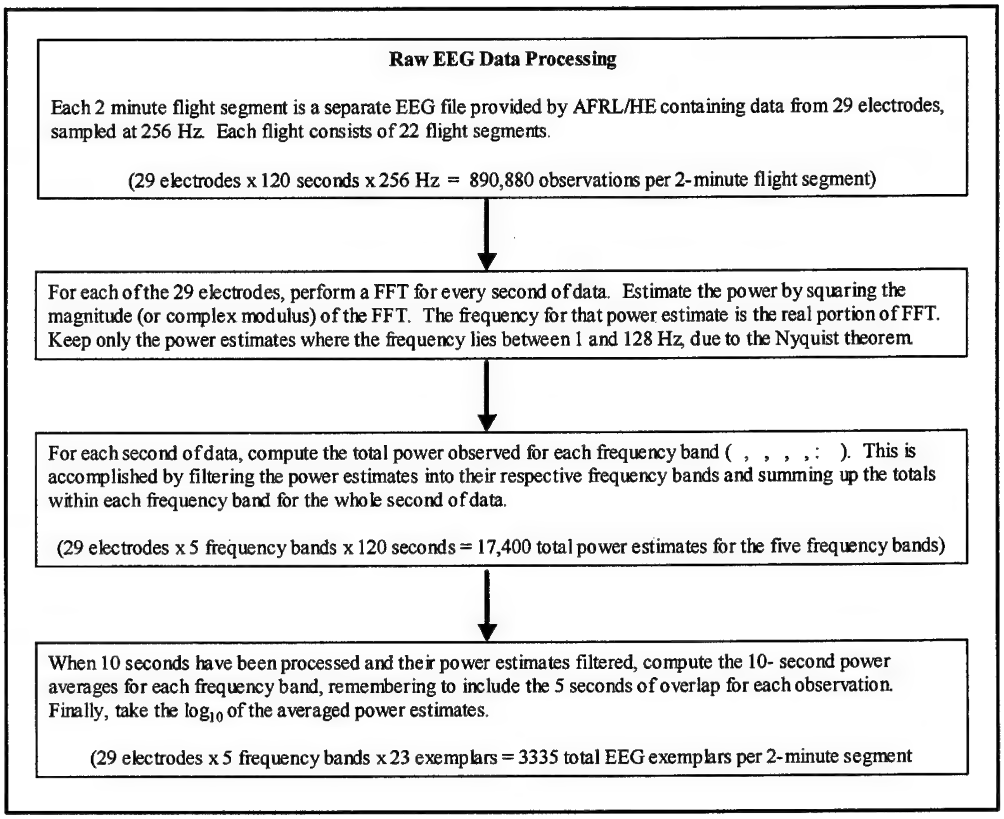


Figure 3-3. Raw EEG Signal from Electrode C3 during Landing Segment

The EEG data, since it is a function of time, has a time dependency associated with it. In order to use the data as a classifier, however, this dependency needs to be removed. To remove this dependency, the raw data is passed through a Fast Fourier Transform (FFT), which is a computationally efficient way of computing Fourier Transforms [8]. The FFT moves the data from the time domain into the frequency domain, which will then allow estimates of power to be computed. According to the Nyquist sampling theorem, estimates for power can only be made for frequencies up to $f_s/2$, where f_s is the sampling frequency [8]. Since the EEG data was collected at 256 Hertz (Hz), the estimates for power can be made up to 128 Hz.

Using macros in both Microsoft Excel and Microsoft Word, a software program was developed to automatically preprocess all the EEG data for one pilot during one flight. The code for this preprocessing is shown in Appendix A. The EEG data preprocessing algorithm can be easily understood by following one second of data from one of the 29 electrodes through the process. The process is depicted in Figure 3-4 below, and it must be repeated 76,560 times per flight in order to build the EEG portion of one data set for one pilot.

First, a FFT is performed over one second of raw EEG data on one of the 29 electrodes. This produces 256 rows of primarily complex numbers, since the data was sampled at 256 Hz. The frequency for each of the rows is then found by looking at the real number portion of the FFT output. Due to the Nyquist theorem mentioned earlier, only the frequencies between 1 and 128 Hz are usable, leaving the other frequencies and rows to be disregarded. For all of the rows whose frequencies fall between 1 and 128 Hz, the absolute value (also known as the complex modulus or magnitude) of each FFT output row is calculated, and this result is squared. A filter then pulls out the real number portion of this squared value, producing an estimate of the power at the associated frequency. At this point, the frequencies are filtered into the five desired frequency bands introduced in Chapter II that lie between 1 and 40 Hz, and all of the rows with power estimates falling into each frequency band are summed for the entire second of data. The sum of these power estimates, separated by frequency band, represents the power estimates for that one second of EEG data at that one electrode.



Raw EEG Data Processing

Each 2 minute flight segment is a separate EEG file provided by AFRL/HE containing data from 29 electrodes, sampled at 256 Hz. Each flight consists of 22 flight segments.

(29 electrodes x 120 seconds x 256 Hz = 890,880 observations per 2-minute flight segment)

For each of the 29 electrodes, perform a FFT for every second of data. Estimate the power by squaring the magnitude (or complex modulus) of the FFT. The frequency for that power estimate is the real portion of FFT. Keep only the power estimates where the frequency lies between 1 and 128 Hz, due to the Nyquist theorem

For each second of data, compute the total power observed for each frequency band (, , , ,:). This is accomplished by filtering the power estimates into their respective frequency bands and summing up the totals within each frequency band for the whole second of data.

(29 electrodes x 5 frequency bands x 120 seconds = 17,400 total power estimates for the five frequency bands)

When 10 seconds have been processed and their power estimates filtered, compute the 10- second power averages for each frequency band, remembering to include the 5 seconds of overlap for each observation. Finally, take the $\log_{10}$ of the averaged power estimates.

(29 electrodes x 5 frequency bands x 23 exemplars = 3335 total EEG exemplars per 2-minute segment

Figure 3-4. Raw EEG Data Preprocessing Chart

Figure 3-5 shows an example of the power estimates found by using this method over a one second interval, broken down into the five different frequency bands. This graph is also known as a periodogram. As shown on the x-axis, only frequencies from 1 - 40 Hz are included in this research effort. Frequencies above 40 Hz are often associated with muscular movements and not mental workload, so they are not processed or analyzed [25]. The vertical lines separate the different frequency bands, and the y-axis identifies the estimated power values, expressed in microvolts^2 (μV^2).

The periodogram allows one to visualize the estimate of power contained in the EEG signal. Unfortunately, periodogram estimates of power obtained from a FFT decomposition often have a large variance that do not decrease even if the sample size is increased [17]. The variance can be reduced, however, by breaking the signal into separate sections and averaging the power across these sections. For example, if each section represents one second of data, then averaging the power over several seconds of data reduces the variance in the resulting power estimates. The more sections the power is averaged over, the lower the variance in the estimates [17].

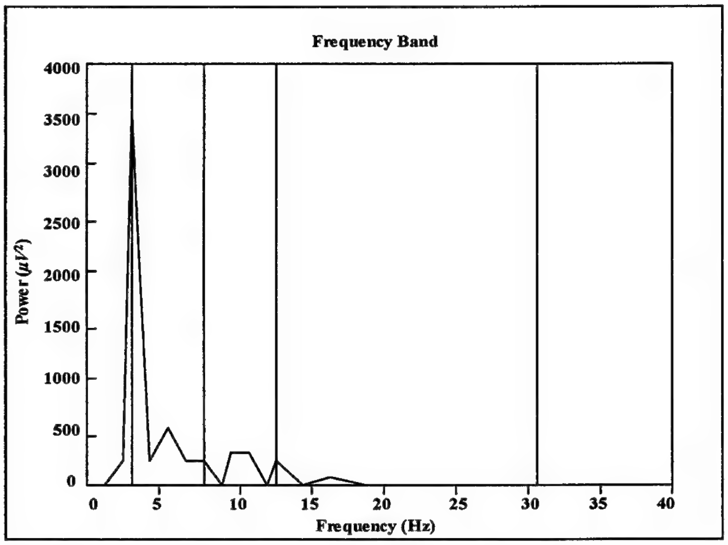


Figure 3-5. Power Estimates by Frequency For One Electrode During One Second

To reduce the variance and smooth the EEG power estimates, all power estimates for each frequency band in this research effort are averaged over a 10-second window that includes 5-second overlaps with the previous observation. Figure 3-6 shows a graph depicting how the observations are built using this overlapping window concept. The overlapping sections are statistically dependent and therefore increase the variance. More sections (i.e. seconds of data) can be used to help alleviate this increase in variance, however 10 sections were found to be adequate in past research [10, 15].

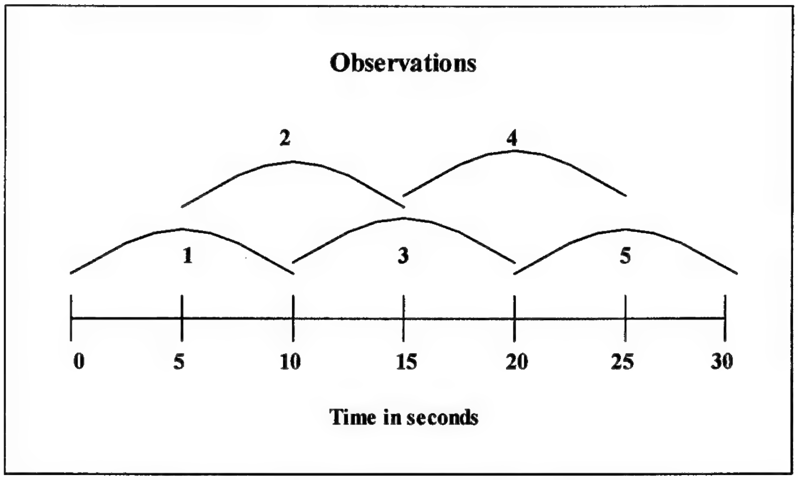


Figure 3-6. Overlapping Window Construction

This overlapping window method produces 12 distinct non-overlapping windows and 11 overlapping windows that are a combination of the distinct non-overlapping windows. The 12 distinct non-overlapping windows are the odd windows shown in Figure 3-6, and the 11 combination overlapping windows are the even windows in the figure. The result is a total of 23 exemplars of averaged power estimates for each two-minute flight segment. Over a 44-minute flight, therefore, a total of 506 exemplars per frequency band are generated for analysis.

The final step in preprocessing the EEG data occurs after averaging the data over each 10-second time window. This step entails scaling the average power estimates using the $\log_{10}$ transformation. An example of a fully processed two-minute flight segment for one electrode is shown in Figure 3-7. Upon completion of this final step, 145 features based upon the EEG data are developed for use in classifying mental workload per 2-minute segment, with 23 exemplars available per node.

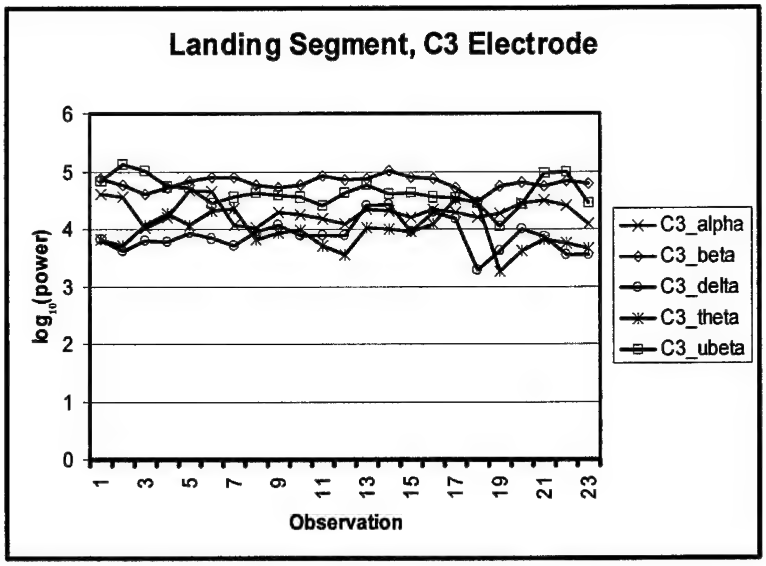


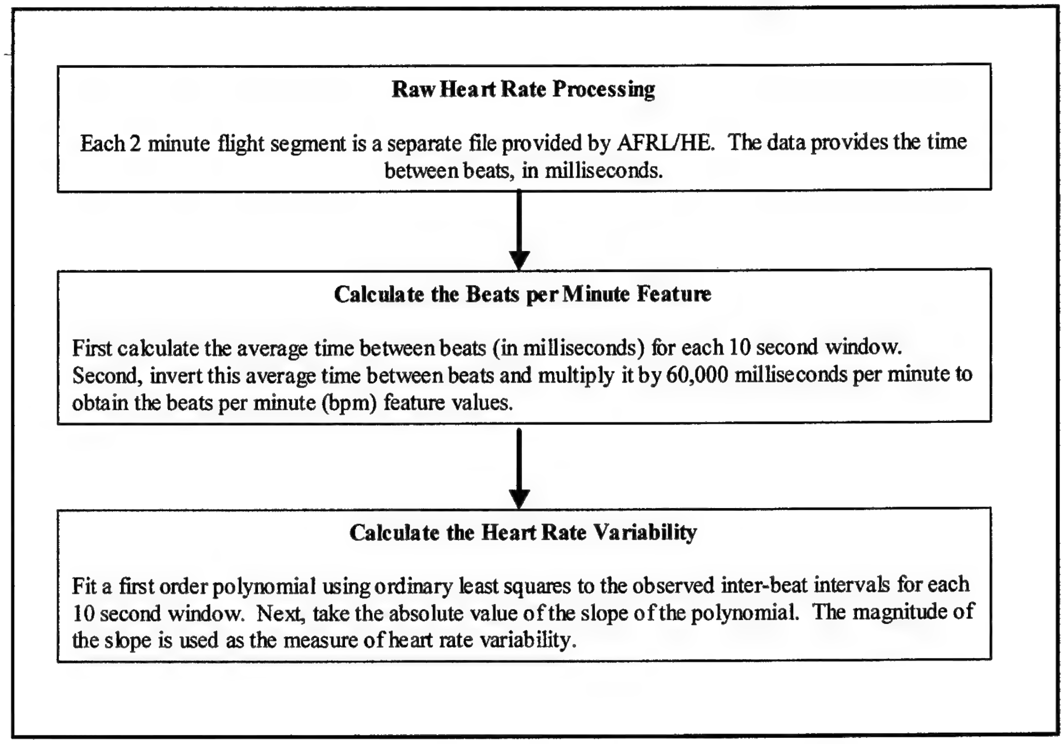
Figure 3-7. Processed EEG Signal

3.4 Physiological Feature Preprocessing

The preprocessing required for the remaining physiological features from the heart, eye, and respiratory files is less involved than the EEG data preprocessing and brings the total to 151 features available for classifying mental workload. To allow EEG and physiological features to be included together within data sets, the same overlapping 10-second window method described in Section 3.3 is employed. This produces 23 exemplars per 2-minute flight segment, as way true for the EEG preprocessing. The same software tool described in Section 3.3 also processes the remaining physiological features described in this section, however only the Microsoft Excel portion is needed to process these remaining features. The software code for the physiological feature preprocessing is included in Appendix A.

3.4.1 Cardiac Measures. The raw heart rate files contain the time between heartbeats, in milliseconds, for each two-minute flight segment. By processing the cardiac files, two different features are developed. The first feature is the heart rate (in beats per minute), and the second feature is the heart rate variability. The heart rate variability is most easily thought of as the rate of increase or decrease in the heart rate over a period of time, which in this case is every ten seconds. Figure 3.8 provides a procedural summary of how the software tool preprocesses the two cardiac measures.

The first step involves computing the average beats per minute. Since the data reflects the time between heartbeats (in milliseconds), the average time between beats for each 10-second window is calculated, and then inverted. After multiplying this result by 60,000 milliseconds per minute, the average beats per minute for each 10-second window is obtained. Figure 3-9 shows an example of a fully processed average beat per minute flight segment.



Raw Heart Rate Processing

Each 2 minute flight segment is a separate file provided by AFRL/HE. The data provides the time between beats, in milliseconds.

Calculate the Beats per Minute Feature

First calculate the average time between beats (in milliseconds) for each 10 second window. Second, invert this average time between beats and multiply it by 60,000 milliseconds per minute to obtain the beats per minute (bpm) feature values.

Calculate the Heart Rate Variability

Fit a first order polynomial using ordinary least squares to the observed inter-beat intervals for each 10 second window. Next, take the absolute value of the slope of the polynomial. The magnitude of the slope is used as the measure of heart rate variability.

Figure 3-8. Raw Cardiac Data Preprocessing Chart

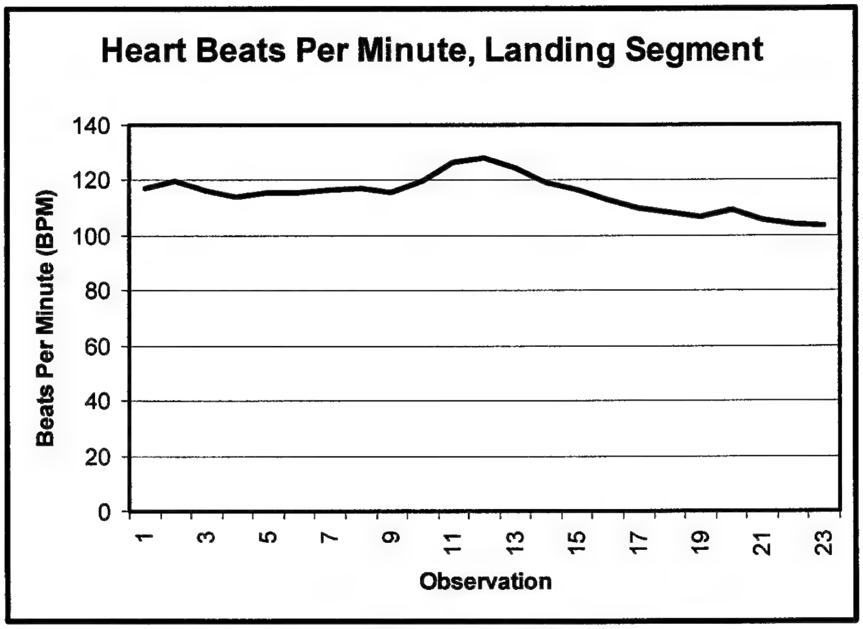


Figure 3-9. Processed Heart Beats Per Minute Feature

The second heart feature is the heart rate variability. To calculate this feature, the software tool performs a first order polynomial fit using ordinary least squares to the time intervals between heartbeats in each 10-second time window. If a heartbeat overlaps a 10-second time window cut-off, then its value is included in the next time window calculation. Upon completion of the polynomial fit, the last part of the cardiac preprocessing occurs. This consists of simply taking the absolute value of the slope from the polynomial fit to estimate the change in heart rate. The magnitude of this slope is used as the measure of heart rate variability. Figure 3-10 shows an example of a fully processed heart rate variability feature for one flight segment.

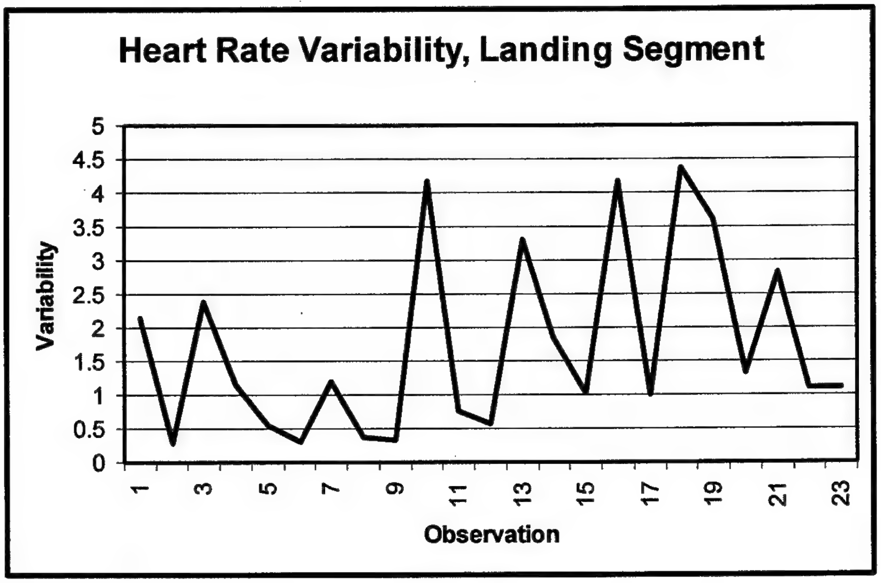
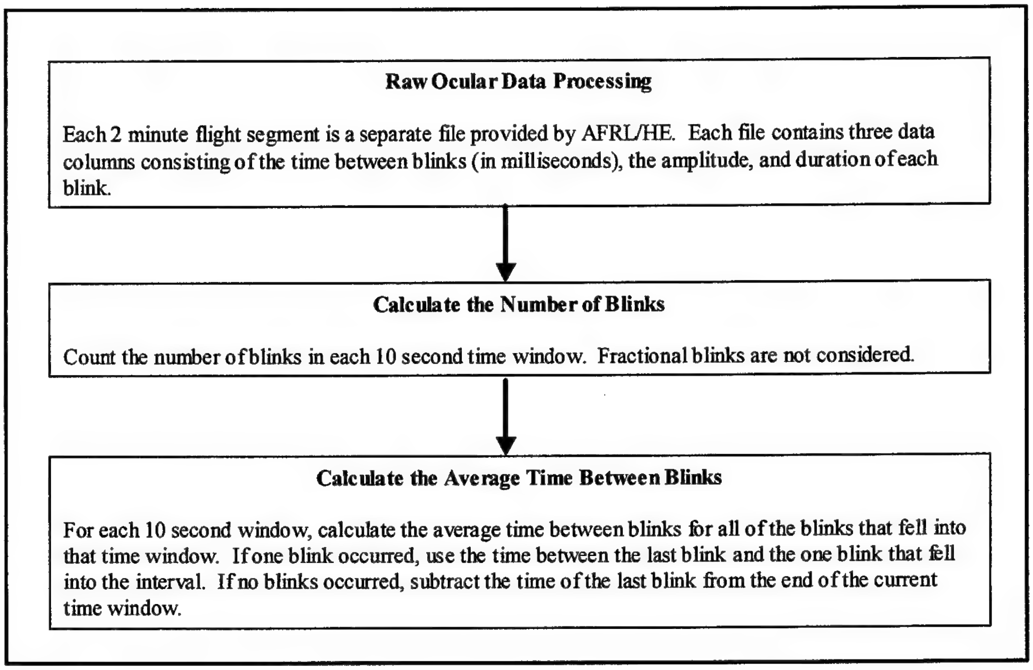


Figure 3-10. Processed Heart Rate Variability Feature

3.4.2 Ocular Measures. Two ocular measures are calculated from the data provided by AFRL/HE, however the raw eye data files contain three different columns of eye data: the blink interval (the time in milliseconds between blinks), the blink amplitude, and the blink duration. The blink duration data is disregarded, and the other two data columns are used to develop the number of blinks per 10-second time window and the average time between blinks. The same software tool introduced in previous sections automatically performs all of the ocular data preprocessing by following the diagram seen in Figure 3.11.



Raw Ocular Data Processing

Each 2 minute flight segment is a separate file provided by AFRL/HE. Each file contains three data columns consisting of the time between blinks (in milliseconds), the amplitude, and duration of each blink.

Calculate the Number of Blinks

Count the number of blinks in each 10 second time window. Fractional blinks are not considered.

Calculate the Average Time Between Blinks

For each 10 second window, calculate the average time between blinks for all of the blinks that fell into that time window. If one blink occurred, use the time between the last blink and the one blink that fell into the interval. If no blinks occurred, subtract the time of the last blink from the end of the current time window.

Figure 3-11. Raw Ocular Data Preprocessing Chart

The first feature, the number of blinks, is quite simple to calculate. It entails counting the number of blinks that fell into each 10-second time window. Fractional blinks are not considered, as they will naturally fall into a future 10-second time window.

The second feature, the average time between blinks, is a more complicated feature to calculate since three scenarios are possible. If multiple blinks fall into a 10-second time window, then the simple average of the time between these blinks is used. On the other hand, if only one blink falls in a 10-second time window, then the time between the last blink and the blink in the current is used. Finally, if no blinks fall into a 10-second time window, then the average time between blinks is determined by subtracting the time of the last blink from the end of the current time window. Figure 3-12 shows a graph of the number of blinks in a 2-minute flight segment, and Figure 3-13 shows a graph of the average time between blinks for the same 2-minute flight segment.

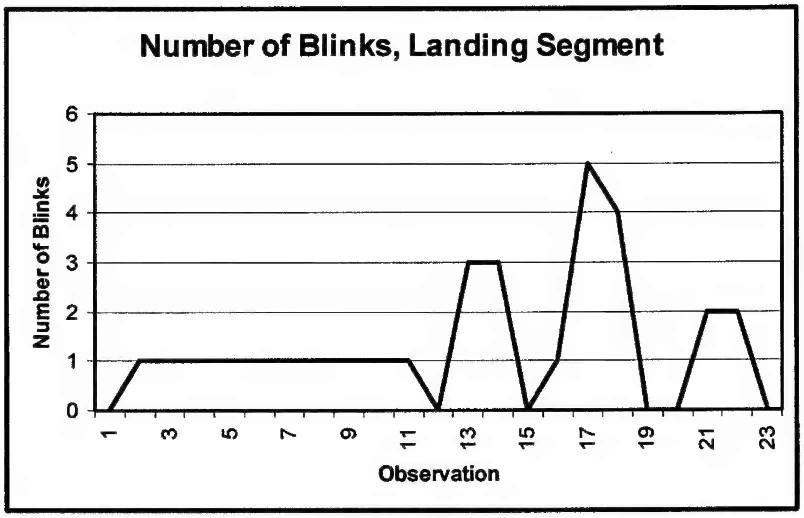


Figure 3-12. Processed Number of Blinks Feature

3.4.3 Respiration Measures. The two respiration features developed from the raw respiration data files are the number of breaths per 10-second time window and the average time between breaths within the time window. The data files provided by AFRL/HE, however, include three data columns. These data columns are: the time

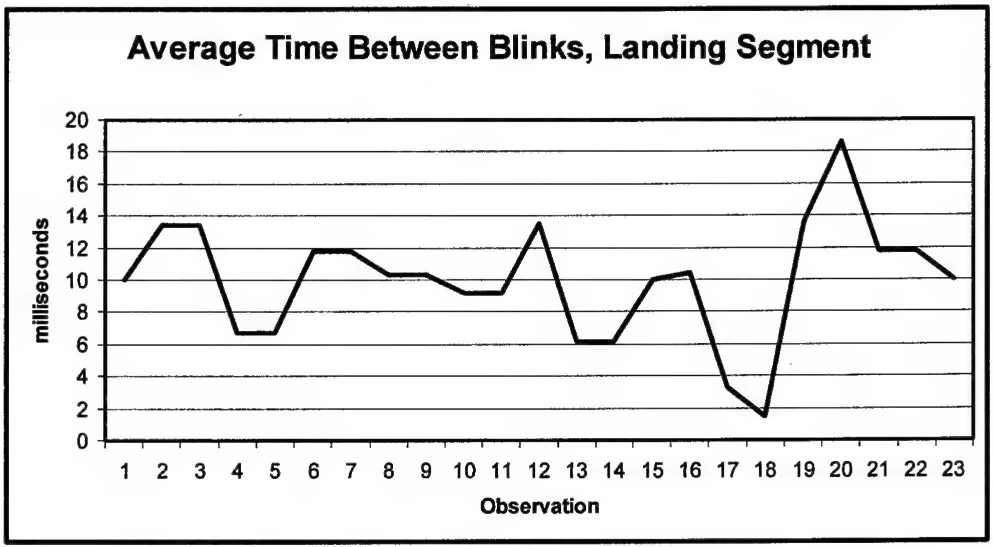
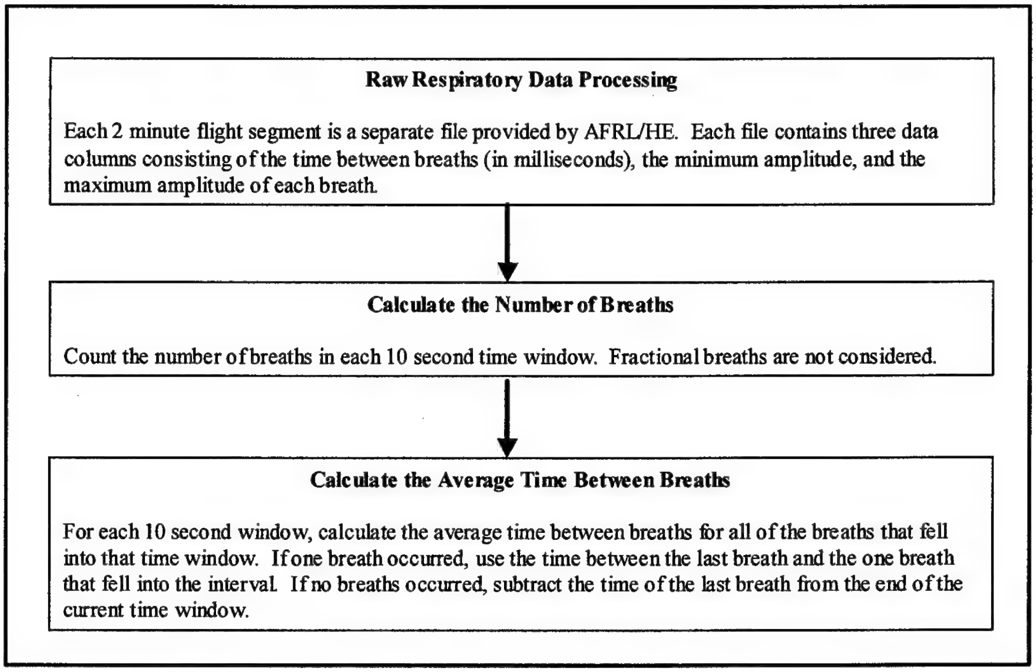


Figure 3-13. Processed Average Time Between Blinks Feature

between breaths (in milliseconds), the minimum breath amplitude, and the maximum breath amplitude. Only the time between breaths data column is used to develop both respiration features, and the preprocessing procedures are identical to those used in preprocessing the ocular features. Figure 3-14 identifies the process to develop these two features, and it is the method used by the software tool to automatically calculate them.

The number of breaths feature is simply the number of breaths that occur in each 10-second time window. Just like the ocular feature procedure, no fractional breaths are included since they will be reflected in future time windows. The average time between breaths feature is found by averaging the time between breaths within a 10-second time window. If only one breath occurs in a time window, then use the time between the last breath and the one breath in the interval. If no breaths occur in a time window, then subtract the time of the last breath from the end of the current time window. Figures 3-15 and 3-16 show examples of these two features for the same 2-minute flight segment.



Raw Respiratory Data Processing

Each 2 minute flight segment is a separate file provided by AFRL/HE. Each file contains three data columns consisting of the time between breaths (in milliseconds), the minimum amplitude, and the maximum amplitude of each breath.

Calculate the Number of Breaths

Count the number of breaths in each 10 second time window. Fractional breaths are not considered.

Calculate the Average Time Between Breaths

For each 10 second window, calculate the average time between breaths for all of the breaths that fell into that time window. If one breath occurred, use the time between the last breath and the one breath that fell into the interval. If no breaths occurred, subtract the time of the last breath from the end of the current time window.

Figure 3-14. Raw Respiratory Data Preprocessing Chart

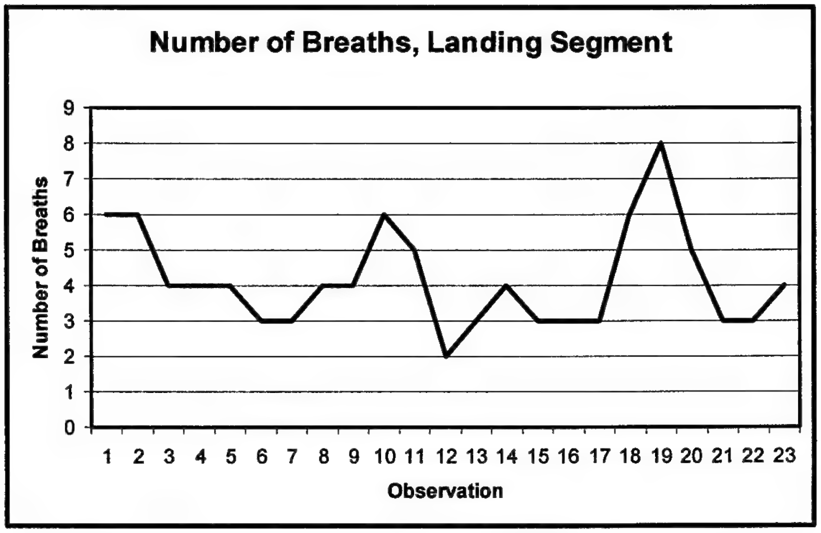


Figure 3-15. Processed Number of Breaths Feature

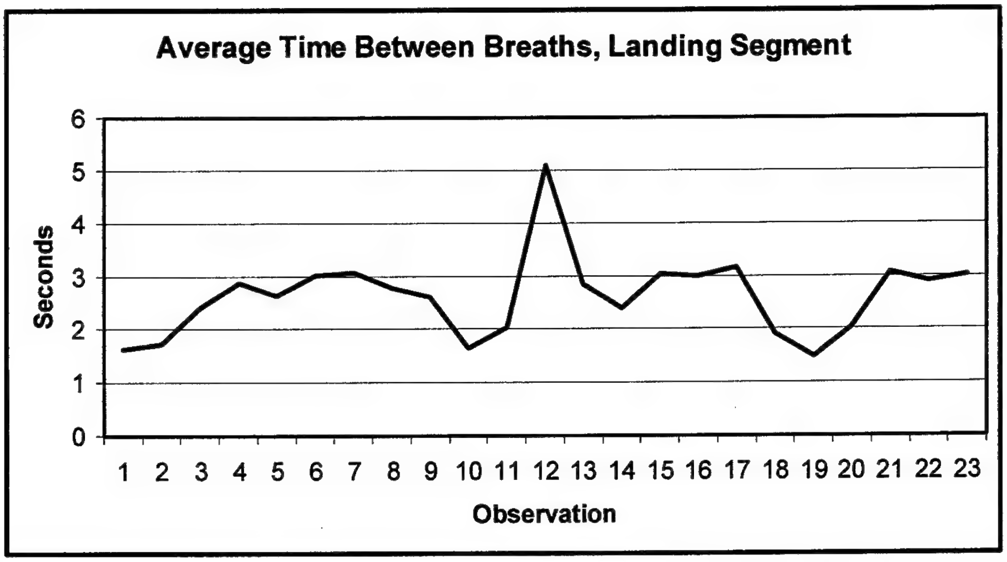


Figure 3-16. Processed Average Time Between Breaths Feature

3.5 Handling Data Gaps

One problem often encountered when using data from real test subjects versus simulated data, is the possibility of having holes or gaps in the data. The data for this experiment had several cases where EEG features were missing for various lengths of time. Most likely this was the result of a loss of contact between the pilot and one of the twenty-nine electrodes. The options available to solve this problem include deleting each feature containing a gap from the data set, or filling the gap with non-zero data. If the first option is chosen and the entire feature is deleted from the data set, fair comparisons of variable sets across pilots or across days would require that the feature be removed from every data set. Should this feature be highly significant in predicting mental workload, then its removal could seriously affect the final selection of the most salient features and possibly the ANN's ability to accurately classify mental workload. If the gap is filled with non-zero data, then a decision must be made concerning how to best accomplish this action without losing the data integrity of the affected features.

The second option seems most appropriate. We decided to keep the affected EEG features with missing data, and fill the gaps with an average value based on the location of the gap. If the gap occurred in the middle of the data set, then the two data points immediately above and below the gap were used to create an average value for filling the gap. If the gap occurred at the end of the data set, then the four data points immediately above the gap were used to create the average value for filling the gap. If the gap occurred at the beginning of the data set, then the four data points immediately following the gap were used to create the average value for filling the gap. The most likely effect of this procedure will be an overall reduction in the total variance observed in each affected feature. We felt that accepting this slight reduction in variance was preferable to the total loss of the feature from the data sets.

3.6 Summary of Processed Features

Once all of the data preprocessing has been accomplished, a total of 151 psychophysiological features are available to the ANN for classifying mental workload. In order to reduce the number of features through the Signal-to-Noise ratio algorithm, one last feature must be added to the data sets. This feature is the noise feature, and it consists of random numbers drawn from a uniform (0, 1) distribution. Binary mental workload values are also added to each row of the data sets, with a 0.0 representing low mental workload and a 1.0 indicating high mental workload. A truncated version of the final input matrix is shown in Table 3-2.

Table 3-2. Truncated Input Feature Matrix

Feature #	**Name**	**Description**	**Units**
1	Workload Level	0 if low, 1 if high	None
2	C3_delta	Power in Band at C3	$\log_{10}(\mu V^2)$
3	C3_theta	Power in Band at C3	$\log_{10}(\mu V^2)$
4	C3_alpha	Power in Band at C3	$\log_{10}(\mu V^2)$
5	C3_beta	Power in : Band at C3	$\log_{10}(\mu V^2)$
6	C3_ultrabeta	Power in Band at C3	$\log_{10}(\mu V^2)$
7	C4_delta	Power in Band at C4	$\log_{10}(\mu V^2)$
8	C4_theta	Power in Band at C4	$\log_{10}(\mu V^2)$
9	C4_alpha	Power in Band at C4	$\log_{10}(\mu V^2)$
10	C4_beta	Power in : Band at C4	$\log_{10}(\mu V^2)$
11	C4_ultrabeta	Power in Band at C4	$\log_{10}(\mu V^2)$
146	Heart_Rrate	Heart Rate	bpm
147	Heart_Variability	Heart Rate Variability	sec per 10-sec
148	Blinks	Number of Eye-Blinks	# blinks per 10-sec
149	Inter_Blink	Inter-blink Interval	seconds
150	Breaths	Number of Breaths	# breaths per 10-sec
151	Inter_Breath	Inter-breath Interval	seconds
152	Noise	Random Uniform (0,1)	none

It is important to note that previous initial data inspections on this data has found that some of the psychophysiological features appear to vary with an increased workload level. Most notably, the heart rate increases, the number of eye blinks decrease, and the number of breaths tend to increase as mental workload increases [10]. Previous research with feature screening has also shown that these features are significant in predicting mental workload, and networks trained with data from one day did, in fact, produce reasonably high classification accuracies when projected onto data from the same day

[10]. Despite this success, a network trained with data from one day did a very poor job of accurately classifying the mental workload for the same pilot on a different day [10].

3.7 Chapter Summary

This chapter addressed how to preprocess the various data files to develop 151 different psychophysiological features for use when classifying mental workload. In the next chapter, the methodology used to classify mental workload will be investigated, and variable selection and reduction efforts will be accomplished. Factor analysis will also be presented to see what additional information and insight can be garnered from the data, and a calibration scheme will be presented.

IV. Methodology

This chapter describes the methodologies used to classify pilot mental workload by means of the processed psychophysiological features described in Chapter III. Following some general methodology information in the first section, the second section is devoted to the initial modeling efforts where the salient features are found in each data set. The third section presents the methodology used for conducting factor analysis and the accompanying exploratory factor analysis. The fourth section addresses different ways to modify the mental workload levels as we explore the possibility that some of the assumptions of this research effort are sources of low classification accuracy. Finally, the fifth section identifies a data calibration scheme that can be applied to the original and modified workload levels, as well as several different training groups.

4.1 General Methodology Information

To highlight some of the subtle changes that occur between several of the methodologies presented in this chapter, and to help avoid confusion, certain sections will be presented with a common table identifying key pieces of information associated with the method in that section. A sample information table, shown in Table 4-1, shows the workload type, the training group set, and identifies whether the data was calibrated following the calibration scheme. The workload type identifies what mental workload levels were used in the training, training-test, and validation data sets. The three possible choices include "original" workload, "modified with high-once-high" workload, and "modified with neither" workload. We will describe each workload type in-turn. The "original" workload designation means that the mental workload levels originally agreed upon by the pilots and the researchers at AFRL/HE were used for the three data sets. Table 4-2 lists the flight segments and these "original" workload levels.

Table 4-1. Sample Information Table

Type of Information	Description
Workload Type	Original
Training Group Set	All flight segments
Data Calibrated?	No

Table 4-2. Original Workload Designations By Flight Segment

Segment #	Flight Segment	Workload Level
1	Baseline 1	1
2	Preflight	1
3	Engine Start	1
4	VFR Takeoff	1
5	VFR Climbout 1	1
6	VFR Cruise	1
7	VFR Airwork	1
8	Approach	1
9	VFR Touch and Go	2
10	VFR Climbout 2	1
11	IFR Airwork	2
12	IFR Cruise	2
13	IFR Hold	2
14	IFR DME Arc	2
15	IFR ILS Tracking	2
16	IFR Missed Approach	2
17	IFR Climbout	1
18	HS Hold	1
19	HS DME Arc	1
20	HS ILS Tracking	2
21	Landing	2
22	Baseline 2	1

Figure 4-1 identifies the subjective levels of mental workload for each flight segment, and the thick horizontal line drawn across the graph establishes these "original" workload designations by separating the low from high mental workload levels.

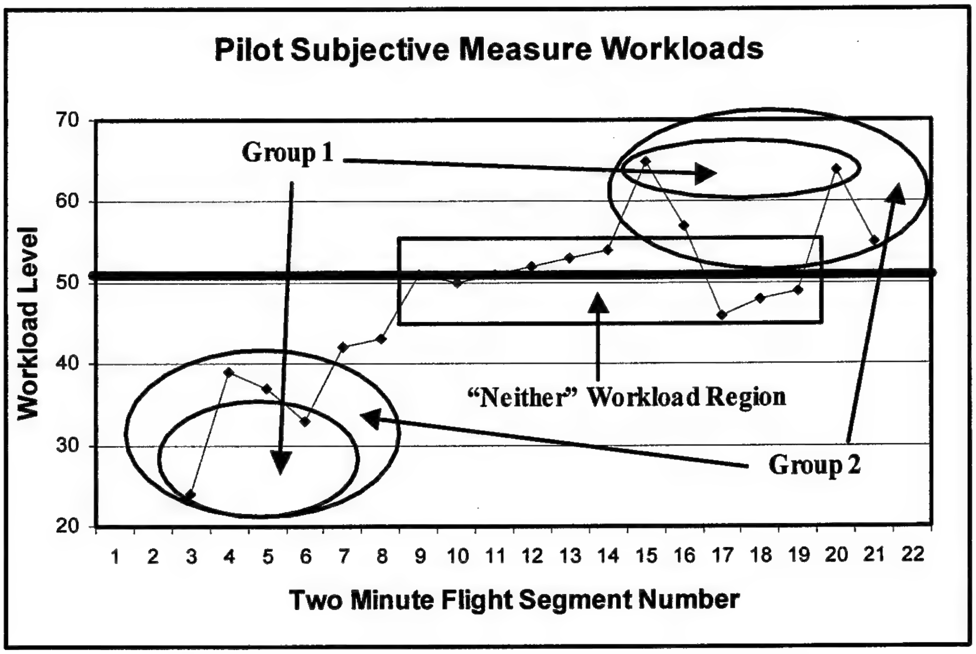


Figure 4-1. Workload Levels and Training Group Sets

The "modified with high-once-high" workload designator, not diagramed in Figure 4-1, means that all flight segments following the first high workload flight segment (segment 9) are changed to high workload, regardless of their original workload levels. This reason for this modified workload method stems from the possibility that after pilots hit high mental workload, their current mental workload level remains affected by either the recent workload increase or their anticipation of future workload increases. As a consequence, regardless of a decrease in the actual current mental workload, it is possible that their brains do not allow them to return to a lower mental workload level. An example where this could occur is a pilot repeatedly performing a difficult maneuver for several minutes using only instruments in poor weather. Following a sharp increase in altitude, visibility improves to several miles and the apparent mental workload level drops. Instead of the pilot's actual mental workload level dropping, it remains elevated because he is still thinking about the difficult maneuvers he recently completed.

The "modified with neither" workload designator takes into account the possibility that there is not a single line separating high from low mental workload, but actually an indifference zone where the mental workload is neither high nor low. Under this workload modification method, the "neither" workload area falls both a little above and below the horizontal line shown on Figure 4-1, and includes the boxed flight segments (flight segments 9 through 14, 17 through 19, and 22).

The training group set in Table 4-1 identifies which flight segments were used to train the network. The three choices are: all flight segments, Group 1, and Group 2. A response of "all flight segments" means that every flight segment was included in the training and training-test data sets. A "Group 1" response identifies that only those flight segments nearest to the extremes (lowest workload and highest workload) are used when training the network. By looking at Figure 4-1, the Group 1 flight segments are included in the smaller circles at both the lower and upper portions of the graph. Flight segments 3, 6, 15, and 20 fall into the Group 1 training group. Similarly, a "Group 2" response identifies that only those flight segments included in Group 2 are used when training the network. Group 2, shown by the two larger circles in Figure 4-1, includes all of the flight segments from Group 1, plus the next two most extreme flight segments for both high and low mental workloads. As a result, the Group 2 flight segments include segments 3 through 6, 15 through 16, and 20 through 21.

One additional point concerning the two training group sets is that each group contains equal numbers of high and low flight segments. This can be an important consideration since networks trained with an overwhelming number of exemplars from one particular class can sometimes achieve a minimum squared error by always classifying exemplars as members of the dominant class, regardless of their true membership class.

The final piece of information in Table 4-1 identifies whether or not the data was calibrated using the calibration scheme prior to training the ANN. This calibration scheme is not presented until Section 4.5.

4.2 Initial MLP Neural Network Modeling Efforts

Upon completing the preprocessing of the psychophysiological data, the next step involves training ANNs to find the most salient features for each pilot on each day. Every neural network for this research effort is built with the same basic architecture and settings so that differences in classification accuracy can be attributed primarily to the selected workload type, training group set, and whether or not the data was calibrated using the calibration scheme. These settings are shown in Table 4-3.

Table 4-3. Basic Network Architecture and Parameter Settings

Architecture or Parameter	**Network Setting**
Training-test data set	Holdout exemplars from training data set found by mod 3, remainder 0
Input Variables	Normalized
Training Rate	0.01
Momentum	0.9
Weight Initialization	-0.1 to 0.1
Termination Rule	Minimum training-test sum of square error

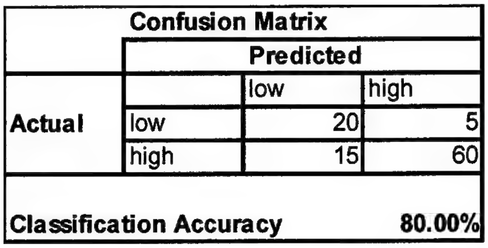
The number of hidden nodes to include in each network often depends on the number of features included in the training data set, and SNNAP suggests a number of nodes accordingly. All of the networks use SNNAP's suggested number of nodes. In addition to these settings, a bias term and two output nodes are included in each model. The two output nodes allow the network to compute probabilities of an exemplar belonging to the high and low workload classes. With these probabilities, network classification accuracy (CA) can be determined using the Equation 4-1, shown below.

$$\mathrm{CA} = \frac{\mathrm{N_{1C}} + \mathrm{N_{2C}}}{n} \tag{4-1}$$

where

- CA is the classification accuracy
- $\mathrm{N_{1C}}$ is the number of exemplars in group 1 classified as group 1
- $\mathrm{N_{2C}}$ is the number of exemplars in group 2 classified as group 2
- n is the total number of exemplars in the test data set

Normally, the maximum CA for a network is found by assigning each exemplar to the group whose output node has a probability greater than or equal to 0.5 [27]. A confusion matrix can then be built using the exemplar assignments by comparing them to the actual classes from which they came. A sample confusion matrix is shown in Figure 4-2. In this example of 100 exemplars, 80 are classified correctly and 20 are classified incorrectly for a CA of 80%. The network incorrectly predicts low 15 times when the actual class membership is high (Type I error), and it incorrectly predicts high 5 times when the actual class membership is low (Type II error).



Confusion Matrix			
	Predicted		
		low	high
Actual	low	20	5
	high	15	60
Classification Accuracy			80.00%

Figure 4-2. Sample Confusion Matrix

Since many of the 151 features in each data set, especially the EEG features, are highly correlated with one another, and partially due to the randomness of the neural network initial weight values, different features can be selected for removal from the same network when identically initialized and trained several times [10]. With the high correlation among the features, any difference in feature selection should have negligible impact on the classification accuracy of the network, and so resolving feature selection differences is unnecessary. The criterion for feature removal is based on low SNRs, as described in the SNR screening method in Chapter II, and Appendix B identifies a process to build the SNRs using SNNAP output. The classification accuracy for several neural networks starts to drop significantly (one or more percent) in several instances when fewer than 36 features remain, prompting the decision to allow no more than 36 features per data set.

4.2.1 SNR Saliency Screening On Individual Day Data Sets. Past feature reduction efforts on this data has found that the number of salient features necessary to obtain high inter-day classification accuracy for individual pilots range from 5 to over 59 [10]. The number of salient features identified below are consistent with these results, however the salient features selected in each data set differ due to reasons provided earlier [10].

The most salient features for each pilot on each day are shown in Tables 4-4 through 4-7, when the entire data set is presented to the network for training. The features are listed alphabetically from left to right across the rows. Pilot 1 has 35 salient features on day 1 and 28 salient features on day 2, while Pilot 4 has 36 salient features on both day 1 and day 2.

Table 4-4. Salient Features for Pilot 1 on Day 1

Variable	**Variable**	**Variable**	**Variable**	**Variable**	**Variable**
Blinks	BPM	C3_theta	C4_alpha	C4_beta	C5_alpha
CZ_theta	C6_delta	F4_delta	F7_alpha	F7_delta	F8_delta
FP1_theta	FP2_delta	FZ_delta	Inter_Blink	O1_alpha	O2_delta
O2_theta	OZ_beta	OZ_ubeta	P10_theta	P10_ubeta	P4_beta
P4_delta	P4_theta	P9_theta	P9_ubeta	PO3_beta	PO3_delta
PO4_alpha	PZ_alpha	PZ_beta	T8_beta	T8_theta	

Table 4-5. Salient Features for Pilot 1 on Day 2

Variable	Variable	Variable	Variable	Variable	Variable
Blinks	BPM	C3_ubeta	C4_alpha	CZ_delta	F7_theta
FP2_ubeta	Hrt_Var	Inter_Blink	Inter_Breath	O1_theta	O2_theta
OZ_ubeta	P3_alpha	P3_beta	P3_delta	P4_theta	P7_delta
P7_theta	P8_beta	PO3_beta	PO3_delta	PO4_beta	PZ_alpha
PZ_theta	T7_beta	T7_ubeta	T8_delta		

Table 4-6. Salient Features for Pilot 4 on Day 1

Variable	Variable	Variable	Variable	Variable	Variable
BPM	C3_ubeta	C6_alpha	C6_beta	C6_ubeta	F3_alpha
F3_beta	F3_delta	F3_ubeta	F7_theta	F8_ubeta	FC1_beta
FC1_theta	FC2_alpha	FP1_beta	FP1_delta	FP2_beta	Hrt_Var
IZ_delta	IZ_ubeta	O1_ubeta	OZ_alpha	OZ_delta	OZ_theta
P10_delta	P10_theta	P3_alpha	P7_theta	P8_beta	P8_theta
P9_delta	PO3_delta	PO4_beta	PZ_theta	T7_ubeta	T8_delta

Table 4-7. Salient Features for Pilot 4 on Day 2

Variable	Variable	Variable	Variable	Variable	Variable
Blinks	BPM	Breaths	C4_delta	C4_theta	C5_alpha
C5_ubeta	CZ_beta	CZ_theta	CZ_ubeta	F3_theta	F3_ubeta
F4_delta	F8_delta	F8_ubeta	FC1_ubeta	FC2_delta	FP1_beta
FP2_alpha	FP2_beta	FZ_theta	Hrt_Var	Inter_Breath	IZ_ubeta
O1_ubeta	P10_theta	P10_ubeta	P4_alpha	P4_beta	P8_beta
P9_alpha	P9_beta	P9_theta	PO3_delta	PO4_ubeta	T7_alpha

4.2.2 SNR Saliency Screening On Multiple Day Data Sets. Since the goal of this research effort is to develop a calibration scheme to classify mental workload across days and pilots, it is valuable to identify which features are important when classifying mental workload for an individual pilot over more than just one day. While this is similar to "peeking" into the future since the second day of data is not available for use when building a classifier based upon the first day of data alone, some insights can be gained by observing the results.

An ANN for an individual pilot is trained only after combining the data sets from both flights into a single large data set. This data set is then randomly split into the training and validation data sets using a 65/35 ratio. Remember that the training-test data set consists of holdout exemplars from the training data set. Tables 4-8 and 4-9 identify the features found most salient in the combined day data sets, where Pilot 1 has 36 salient features and Pilot 4 has 6 salient features.

Table 4-8. Salient Features for Pilot 1 Over Both Days

Variable	**Variable**	**Variable**	**Variable**	**Variable**	**Variable**
Blinks	BPM	Breaths	C3_theta	C4_alpha	C4_delta
C5_ubeta	C6_delta	CZ_theta	F3_alpha	F4_beta	FC1_ubeta
FC2_theta	FP1_alpha	FP1_beta	FZ_ubeta	Inter_Blink	IZ_beta
O1_theta	O1_ubeta	O2_delta	OZ_beta	OZ_theta	P10_beta
P3_beta	P3_theta	P4_delta	P4_theta	PO3_alpha	PO3_beta
PO3_delta	PO4_theta	PZ_alpha	PZ_beta	PZ_ubeta	T7_theta

Table 4-9. Salient Features for Pilot 4 Over Both Days

Variable	**Variable**
BPM	C4_ubeta
F8_ubeta	FC1_alpha
Hrt_Var	P10_ubeta

4.3 Factor Analysis

Factor analysis is based on the idea that the set of all features is explained by a smaller set of underlying factors. In the case of classifying mental workload, even though there are 151 different features, there may be a relatively small number of factors that drive these variables. The way these features are split into the different factors is derived from the variance associated with each feature. Factor analysis assumes that some of the feature variance is due to a common variance due to the factors, and the remainder is uniquely tied to the specific feature [4]. By performing factor analysis, the researcher hopes to identify and interpret the underlying factors to provide greater insight into the problem. A more thorough review of the concepts and mathematics behind factor analysis is found in [4] and [10].

To perform factor analysis, the salient features in each data set from Sections 4.1.1 and 4.1.2 are placed into the statistical software program *SAS JMP*. A separate scree plot is then built in Microsoft Excel using the eigenvalues from each data set, showing the relative size of the different eigenvalues compared to one another. The scree line helps determine how many eigenvalues to keep by establishing the number of factors to rotate using the varimax procedure in *SAS JMP*. Figure 4-3 shows a sample scree plot with scree line drawn on it. Since the scree line falls above the sixth eigenvalue and crosses the top of the fifth eigenvalue, choosing to keep the first four eigenvalues would likely result in an appropriate number of factors to rotate.

The output of the varimax procedure is a factor loadings matrix, and this matrix is used to determine the feature-to-factor assignments. This is accomplished by assigning each feature to the factor with the largest absolute value factor loading for that particular feature. We are able to make these assignments because we have already normalized the input data. Once all the features are assigned to the factors, we eliminate those factors with no features assigned to them, and then attempt to interpret the remaining factors.

Factors are not normally eliminated in factor analysis, however in our analysis we are trying to reduce the number of factors to interpret.

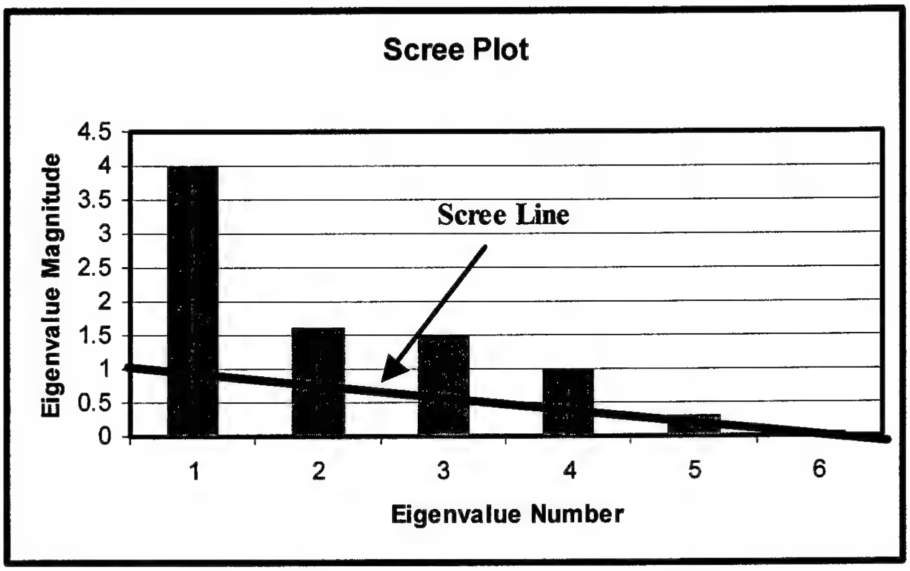


Figure 4-3. Sample Scree Plot and Scree Line

4.3.1 Preliminary Results. A review of the eigenvalues across several of the data sets reveals that the first eigenvalue represents approximately 15% of the total variation in the features, leaving the other eigenvalues to each explain only 3-4% of the remaining variation.

In order to capture a high degree of the total feature variation in these data sets, a large number of factors should be kept. Keeping too many factors does not help reduce the dimensionality of the problem, and therefore limits the effectiveness of performing factor analysis. Keeping too few factors results in low factor loadings matrix values, making it difficult to determine which variables are really correlated to which factor, and also leads to difficulties with factor interpretation. By deciding to set the maximum number of factors to twenty, sufficiently high factor loadings matrix values are produced, and it allows for some useful groupings of features within and across the factors. Table 4-10 identifies the number of factors rotated for each data set.

The decision to limit the number of factors to twenty enables some interpretation of the factors, and more importantly, it highlights key features within each factor that can be explored as we look for patterns to exploit. With the relatively large number of factors for each data set, most of the factors end up being associated with only a few of the features. This makes factor interpretation somewhat easier given that brain researchers have identified certain areas of the brain are associated with certain functions.

Table 4-10. Number of Rotated Factors for Each Data Set

Data Set	Number of Rotated Factors
Pilot 1, Day 1	20
Pilot 1, Day 2	15
Pilot 4, Day 1	20
Pilot 4, Day 2	20
Pilot 1, Mix of both days	20
Pilot 4, Mix of both days	3

A factor with only one feature assigned to it can be interpreted as being related to the function associated with that feature. For a more in-depth interpretation and analysis of the individual factors, reference the work by East [10]. Factor interpretation at this level, however, does not appear to provide direct insight into the research problem, and so an exploratory factor analysis is performed.

4.3.2 Exploratory Factor Analysis. Exploratory factor analysis for this research effort consists of two different activities. The first activity involves compiling the factor results from Section 4.3.1 in different ways to find patterns among the factors. The second activity uses graphs of the key feature-to-factor assignments to find patterns that emerge within the data as mental workload varies.

To identify factor commonalities across pilots and across days, three different compilation methods are employed. The first method involves combining all of the feature-to-factor assignments across the data sets, grouping them by specific feature. The second method groups these feature-to-factor assignments by EEG node, which means dropping the five frequencies associated with each EEG node. The third method groups these feature-to-factor assignments by frequency, which means dropping the EEG node identifiers. A sample of the first grouping method is shown in Table 4-11, and the complete results of the second and third grouping methods are shown in Tables 4-12 and 4-13. The letter "A" indicates the results when using the data set for Pilot 1 on day 1; "B" indicates Pilot 1 on day 2; "X" indicates Pilot 4 on day 1; "Y" indicates Pilot 4 on day 2; "1" indicates Pilot 1 over both days of data; and "4" indicates Pilot 4 over both days of data.

The first two methods of grouping the data do not appear to produce any meaningful patterns. The first method results in the identification of nearly all 151 features associated with one or more of the factors. While the EEG features are evenly spread over the factors, the physiological features are grouped rather tightly in the first six factors across the different data sets. In particular, the second factor shows a high concentration of the physiological features, with the ocular and heart features dominating the factor. The loadings for this factor are shown in Appendix C. Since the physiological features align with the first few factors, this means they likely represent a larger portion of the total variation in the data sets than many of the other features identified in later factors. The second method of grouping produces more desirable clusters of features and factors, but even at this higher level of clustering it is difficult to make sense of the results. The third method of factor grouping produces some interesting results worthy of additional discussion. We address this in Chapter V.

Table 4-11. Partial Feature-to-factor Assignments Grouped By Feature

	Factor Number										
Feature	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Blinks		A, 1									
BPM		B	A, X	1	Y	4					
Breaths		Y		1							
Hrt_Var	4	B	X		Y						
Inter_Blink		A, 1		B							
Inter_Breath		B, Y									
C3_alpha	X										
C3_beta											
C3_delta											
C3_theta				A			1				
C3_ubeta			X			B					
C4_alpha	1		A		B						
C4_beta											
C4_delta					1	Y					
C4_theta								Y			
C4_ubeta	4										
C5_alpha							Y		A		
C5_beta											
C5_delta											
C5_theta											
C5_ubeta									1		Y
C6_alpha											
C6_beta						X					
C6_delta					1		A				
C6_theta											
C6_ubeta				X							
CZ_alpha											
CZ_beta			Y								
CZ_delta			B								
CZ_theta					1			A			Y
CZ_ubeta	Y										

Table 4-12. Feature-to-factor Assignments Grouped By EEG Node

	Factor Number										
Combined Feature	1	2	3	4	5	6	7	8	9	10	11
Blinks		A, 1									
BPM		B	A, X	1	Y	4					
Breaths		Y		1							
Hrt_Var	4	B	X		Y						
Inter_Blink		A, 1		B							
Inter_Breath		B, Y									
C3	X		X	A		B	1				
C4	1, 4		A		B, 1	Y		Y			
C5							Y		A, 1		Y
C6				X	1	X	A				
CZ	Y		Y, B		1			A			Y
F3				X, X	X, X	Y	1		Y		
F4			Y, 1						A		
F7	A, X						B		A		
F8				4	A	X, Y				Y	
FC1			4, 1			Y	X, X				
FC2							X	1		Y	
FP1				X	1	1	X, Y	A			
FP2				A, B, X			Y	Y			
FZ			1		A				Y		
IZ		X		Y, 1	X						
O1	1, 1	X	A, Y		B						
O2	1		B	A			A				
OZ	1, B	X		X	X	A		A, 1			
P10	Y				X	X, 1, A	A		Y		
P3	B			B		X, B, 1		1			
P4	A, Y					A, B		A, 1	1	Y	
P7					B, B		X				
P8	Y	X				B, X	X				
P9			A, A	Y			X			Y	Y
PO3	A, X		B, Y	A			B, 1		1	1	
PO4	B					X	1	A			
PZ	A, A, 1				X		B, B	1		1	
T7	X					B	B		1		Y
T8				A	A	B, X					

The next step in exploratory factor analysis involves generating and analyzing graphs. A graph is made for each feature-to-factor association within the different data sets, representing the most important features across the factors. By generating these graphs, we hope to discover that some features form a pattern with the changing levels of mental workload.

Table 4-13. Grouping of Feature-to-factor Assignments By Frequency

	Factor Number										
Combined Feature	1	2	3	4	5	6	7	8	9	10	11
Blinks		A, 1	B								
BPM	4	B	X, 1		Y						
Breaths		Y	1								
Hrt_Var	4	B	X		Y						
Inter_Blink		A, 1	B								
Inter_Breath		B, Y									
Alpha	X		A, Y, 4	B		Y	A, B, 1	A, X, Y			A, X
Beta	A, A, A, B, B, B, X, Y, Y, 1, 1	X, X	Y	Y, A	1, 1	B, 1, 1, 1	A, X, Y	A, Y, 1			
Delta	A, B			B, X, Y	A, X	A, B, X	A, B, X, Y	A	B, Y, Y, 1	X	B, 1
Theta		X	Y	A, B, 1	B		A, 1	B, 1	A, X, X Y, 1	A, B, Y, 1	A, X, 1
Ultrabeta	X, X, Y, 1, 1	X, 4, 4, 4	Y	Y		Y		1	Y	B, Y	A

As one might expect, most of these graphs reveal no discernible patterns across the mental workload levels. A few graphs, however, do show some interesting apparent patterns. The most noticeable pattern for Pilot 1 on day 1 is found in the interblink feature, shown in Figure 4-4. The solid line at the bottom of the graph indicates high workload levels. We notice a definite increase in the value and variation of the feature as the mental workload level increases from low to high. The only other feature for Pilot 1 on day 1 that exhibits a consistent pattern following changes in mental workload level is the number of blinks feature, shown in Figure 4-5. This feature appears to decrease during periods of higher mental workload. For easier comparison, all four ocular and cardiac features are placed together on one graph for each pilot and day in Appendix D. Artificial biases are added to separate the data on many of the graphs. Upon inspecting Figures D-1 and D-2, one notices the other features for Pilot 1 also vary over time and mental workload levels, but they do not vary consistently like the interblink and number of blinks features. In Figure D-1, for instance, the heart BPM feature increases during the

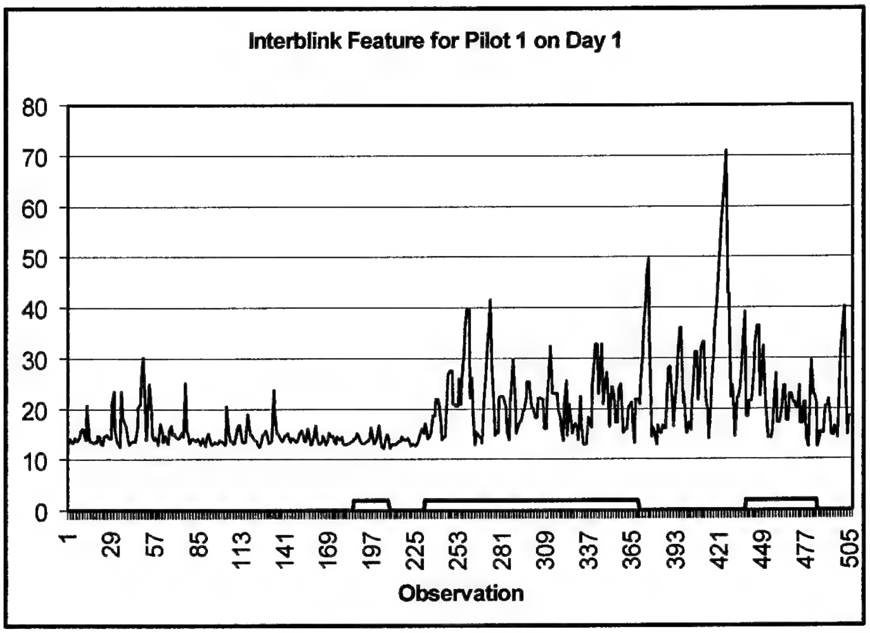


Figure 4-4. Interblink Feature for Pilot 1 on Day 1

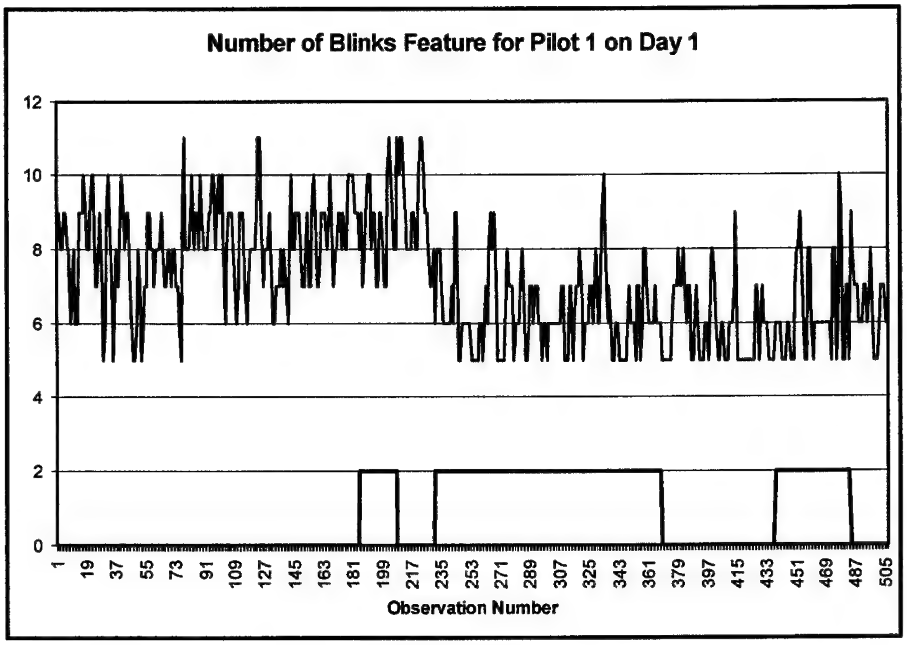


Figure 4-5. Number of Blinks Feature for Pilot 1 on Day 1

first and last periods of higher workload, however it drops during the middle period of higher workload. The heart variability feature follows the same inconsistent pattern as the BPM feature, except it drops during the first and last periods of higher workload and does not change during the middle period.

Inspecting the graph for Pilot 1 on day 2, shown in Figure D-2, reveals similar feature changes to those seen in Figures 4-4 and D-1. Both the number of blinks and interblink features exhibit the same patterns with relation to changes in mental workload. These patterns, however, are not as dramatic as seen on day 1. For instance, the amount of variability in the interblink feature, while certainly higher during periods of greater mental workload, is definitely not as variable as seen on day 1. Perhaps this decrease in variability is due to the learning curve effect caused by the identical flight path and same mental demands being repeated on the second day of the experiment. The increased familiarity possibly allows Pilot 1 on day 2 to lower the visual concentration requirements necessary to execute the same maneuvers performed on day 1. Besides these two features, a search of the remaining features for consistent workload patterns on day 2 reveals no new discoveries.

Similar graphs built using the same features from Pilot 4 on days 1 and 2 reveal surprising different patterns as mental workload varies. The complete graphs with all four features are shown in Figures D-3 and D-4. Unlike Pilot 1, Pilot 4's heart BPM feature rises during periods of higher workload and stays at an overall increased level throughout the higher workload periods. Furthermore, there is a visible decrease in the heart variability feature. Figures 4-6 and 4-7 show Pilot 4's heart BPM and heart variability features for day 1, respectively. A review of the remaining key feature-to-factor assignments for Pilot 4 over the two days, including the EEG and breathing features, reveals no other consistent patterns.

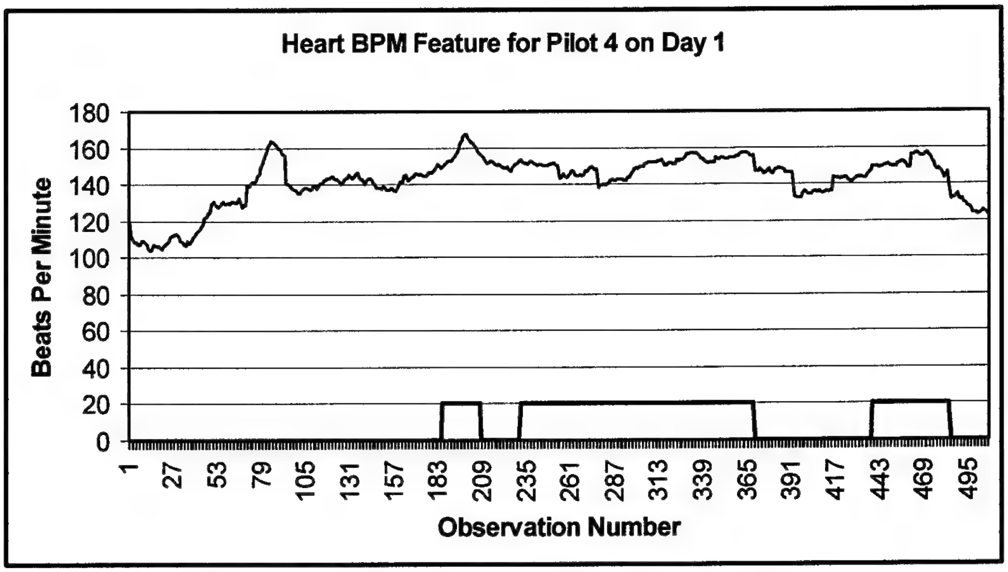


Figure 4-6. Heart BPM Feature for Pilot 4 on Day 1

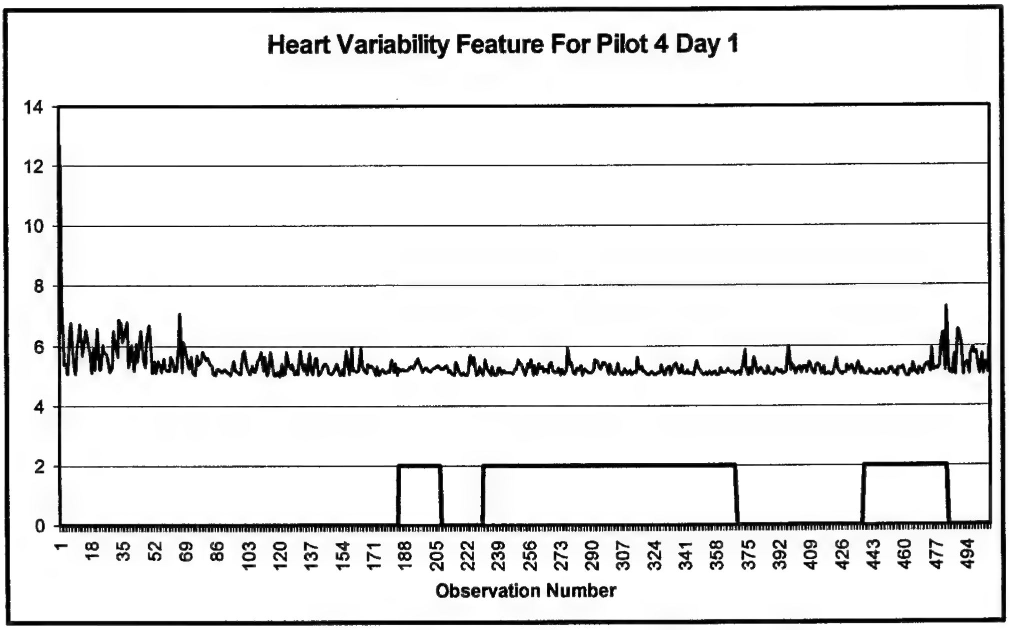


Figure 4-7. Heart Variability Feature for Pilot 4 on Day 1

The different patterns in the psychophysiological features for Pilots 1 and 4 show that the pilots react differently under high workload conditions. Both pilots have two features that reveal patterns with changes to mental workload, but the features are different for each pilot. Furthermore, we notice features not exhibiting patterns for one pilot while exhibiting patterns for the other pilot look like noise features. For example, the graphs for Pilot 1 show decreases in the number of blinks feature and increases in the interblink feature while for Pilot 4 they appear more like noise features. Similarly, the graphs for Pilot 4 show decreases in the heart variability feature and increases in the heart BPM feature while for Pilot 1 they appear as noise features.

From the exploratory factor analysis, we find that Pilots 1 and 4 each have two features that consistently show patterns with the changes in mental workload. We also find that features not containing patterns appear similar to noise features. These discoveries present a new avenue of research for exploitation, discussed in greater detail in Section 4.5.

4.4 Modified Workload Methodologies and Network Training

As mentioned in Section 4.1, several different modifications are made to the original mental workload levels. The reason for these modifications lies in challenging some of the significant assumptions used in this research effort, as discussed in Section 3.1. These assumptions revolve around how accurately the flight segments are classified by workload level, as well as the assumption of instantaneous transitions between varying levels of workload. By modifying the workload levels, the magnitude of these assumptions can be quantified.

4.4.1 Details of the "High-Once-High" Workload Method. Revisiting the first workload modification to the "original" workload levels discussed earlier involves keeping the workload level high once the low/high workload threshold is crossed. The threshold that separates low and high mental workload remains unchanged from the "original" workload levels shown in Figure 4-1. Flight segment 9 first crosses the high/low threshold, and every flight segment after 9 is now reclassified as high workload, except segment 22. Segment 22 remains low because the flight has ended in the experiment and the pilot is sitting stationary on the ground after landing the aircraft.

With the modified workload levels reflected in adjusted data sets for the pilots, all of the ANNs are built and trained. Table 4-14 summarizes the key pieces of information associated with this section. All of the other network settings remain constant.

Table 4-14. Modified Workload Information Table For High-Once-High Method

Type of Information	**Description**
Workload Type	Modified with high-once-high
Training Group Set	All flight segments
Data Calibrated?	No

4.4.2 Details of the "High", "Low", and "Neither" Workload Method. The second modification to the original mental workload levels allows for an indifference zone separating high from low mental workload by including a "neither" workload category. The flight segments that fall into this category, shown back in Figure 4-2, include segments 9 through 14, 17 through 19, and 22. Another modification that is made to the training (and training-test) data sets incorporates the Group 1 and Group 2 training groups mentioned in Section 4.1. As a result, two new training (and training-test) data sets are made from each flight.

ANNs using these modified workload levels and training groups are built and trained. Table 4-15 summarizes the key pieces of information associated with this section. All other network settings remain constant.

Table 4-15. Modified Workload Information Table For “High”, “Low”, “Neither”

Type of Information	Description
Workload Type	Modified with “High”, “Low”, “Neither”
Training Group Set(s)	Groups 1 and 2
Data Calibrated?	No

4.5 Data Calibration Methodology and Network Training

The consistent patterns found in the mental workload data through exploratory factor analysis introduce the possibility of pattern exploitation. If a calibration scheme can be developed that highlights these patterns to an ANN, then mental workload classification accuracy might be improved. Once a calibration scheme is established, one or more new features incorporating the scheme could be used for training the ANNs.

To determine which features to include in the calibration scheme, the features in Section 4.2 identified as most salient in the different data sets are compiled using a five step process. First, the salient features from all of the single flight data sets are combined and then sorted alphabetically. Second, the features that appear more than once are noted in a separate list along with the number of times they appear. The list with features that appear more than once is List #1. Third, the features found most salient across both days per individual pilot are combined and sorted; any features that appear more than once are also noted. This is List #2. Fourth, Lists #1 and #2 are compared and features found in both lists are noted. Fifth, the features that show consistent patterns from the exploratory factor analysis are noted. The features that appear on both lists and show consistent patterns should be included in the calibration scheme. Table 4-16 identifies the results of this process using all of the features identified in Section 4.2. A review of Table 4-16 shows that only four features meet all of the criteria for inclusion in the calibration scheme: eye blinks, heart BPM, heart variability, and interblink.

Following the same process listed above using only the top 10 and 15 features per data, instead of the top 36, produces nearly identical results. The top 10 and 15 features are identified based upon their high ranking of the SNR ratios. Features not identified more than once are not included in the tables. Tables 4-17 and 4-18 show how few of the features repeatedly rank as most important across the two pilots and days. This additional information supports the decision to include the four physiological features listed above in the calibration scheme.

Since the purpose of the calibration scheme is to highlight consistent patterns in the data to the ANN, a linear combination of the features is proposed. The intent is to combine the features in such a way that the sum increases dramatically when approaching high mental workload and drops dramatically when approaching low mental workload.

Table 4-16. Feature Determination for Calibration Scheme

Feature Name	# Times Identified in Individual Flights	# Times Identified Across Days	Consistent Pattern in Data?
Blinks	3	1	Y
BPM	4	2	Y
C3_ubeta	2		
C4_alpha	2	1	
C5_alpha	2		
CZ_theta	2	1	
F3_ubeta	2		
F4_delta	2		
F7_theta	2		
F8_delta	2		
F8_ubeta	2	1	
FP1_beta	2	1	
FP2_beta	2		
Heart Variability	3	1	Y
Interblink	2	1	Y
Interbreath	2		
IZ_ubeta	2		
01_ubeta	2	1	
02_theta	2		
0Z_ubeta	2		
P10_theta	3		
P10_ubeta	2	1	
P3_alpha	2		
P4_beta	2		
P4_theta	2	1	
P7_theta	2		
P8_beta	3		
P9_theta	2		
P03_beta	2	1	
P03_delta	4	1	
P04_beta	2		
PZ_alpha	2	1	
PZ_theta	2		
T7_ubeta	2		
T8_delta	2		

Table 4-17. Top 15 Features Across Pilots and Days

Feature Name	# Times Identified in Individual Flights	# Times Identified Across Days	Consistent Pattern in Data?
Blinks	3	1	Y
BPM	4	2	Y
Interblink	2	1	Y
01_ubeta	2	1	
0Z_ubeta	2		
P8_beta	2		
P03_beta	2	1	
T8_delta	2		

Table 4-18. Top 10 Features Across Pilots and Days

Feature Name	# Times Identified in Individual Flights	# Times Identified Across Days	Consistent Pattern in Data?
Blinks	3	1	Y
BPM	4	2	Y
Interblink	2	1	Y
T8_delta	2		

This might allow the ANN to notice the changes in mental workload more readily since the patterns for each of the features are less distinct individually. Following this concept, the features that are shown to drop when mental workload increases are subtracted from the linear combination, and the features that are shown to increase when mental workload increases are added to the linear combination. The proposed linear combination calibration scheme using standardized data is shown in Equation 4-2. Standardizing each feature is necessary since the features contain different units and are of different magnitudes.

$$\text{New\_1} = -\text{Heart\_Variability}_{SD} + \text{BPM}_{SD} - \text{Blinks}_{SD} + \text{Inter\_Blink}_{SD} \qquad (4\text{-}2)$$

where SD stands for standardized data with a mean of zero and a variance of one. The new feature, labeled New_1, replaces the four natural features when training the ANN. Figure 4-8 shows what this linear combination of features looks like for Pilot 1 on day 1, and it can be compared to Figure D-1 that shows the natural features prior to the linear combination. An artificial bias is added to separate the workload level line from the new feature. In Figure 4-8, the New_1 feature shows an overall increase during periods of higher mental workload and an overall decrease during periods of lower mental workload. We also notice that despite the overall desired movement in the new feature to changes in mental workload, there is a large amount of variability in the linear combination at any given mental workload level.

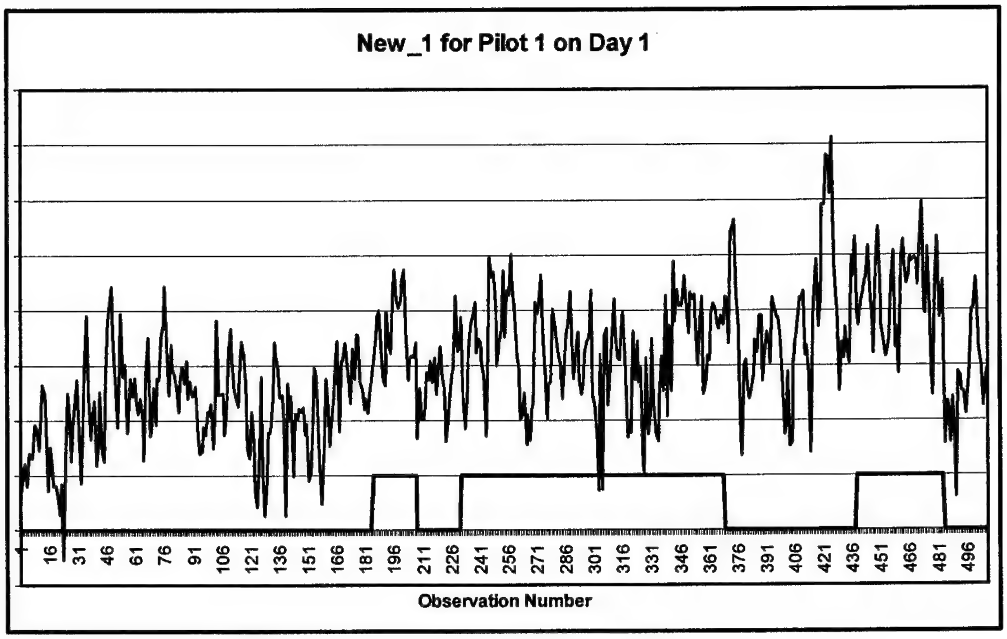


Figure 4-8. Linear Combination of Features for Pilot 1 on Day 1

In order to smooth this variability, three moving averages of New_1 are added to complete the new set of features in the calibration scheme. The lengths of the moving averages are 30, 60, and 120 seconds, and are labeled New_30, New_60, and New_120. With the addition of the moving averages, the four features that comprise the calibration scheme now include New_1, New_30, New_60, and New_120. Figure 4-9 shows the three moving averages for Pilot 1 on day 1. An artificial bias is added to separate the features. As one would expect, the addition of the moving averages smoothes the widely fluctuating New_1 feature. In particular, notice how the New_120 feature generally matches the changes in mental workload.

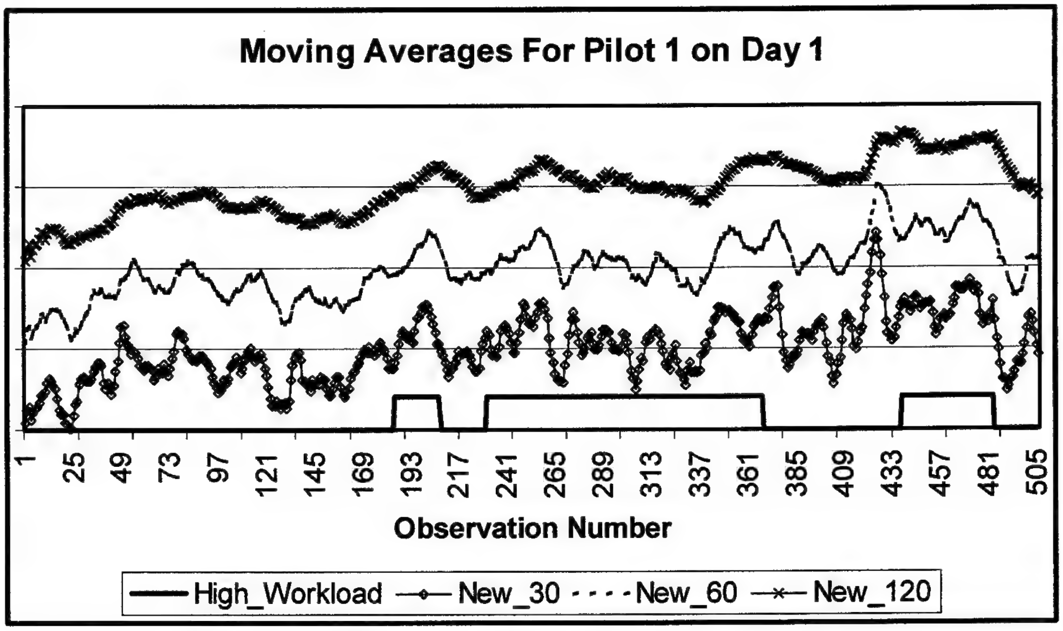


Figure 4-9. Moving Averages for Pilot 1 on Day 1

Figure 4-10 shows the New_120 feature for both pilots over both days to help identify whether or not this New_120 feature matches the changes in mental workload for the other data sets as well. From looking at the figure, it appears that the New_120 feature does generally reflect the mental workload level across pilots and across days. Despite containing greater variability than the New_120 feature shown in Figure 4-10, the other moving average features for each data set also show the same desirable trait.

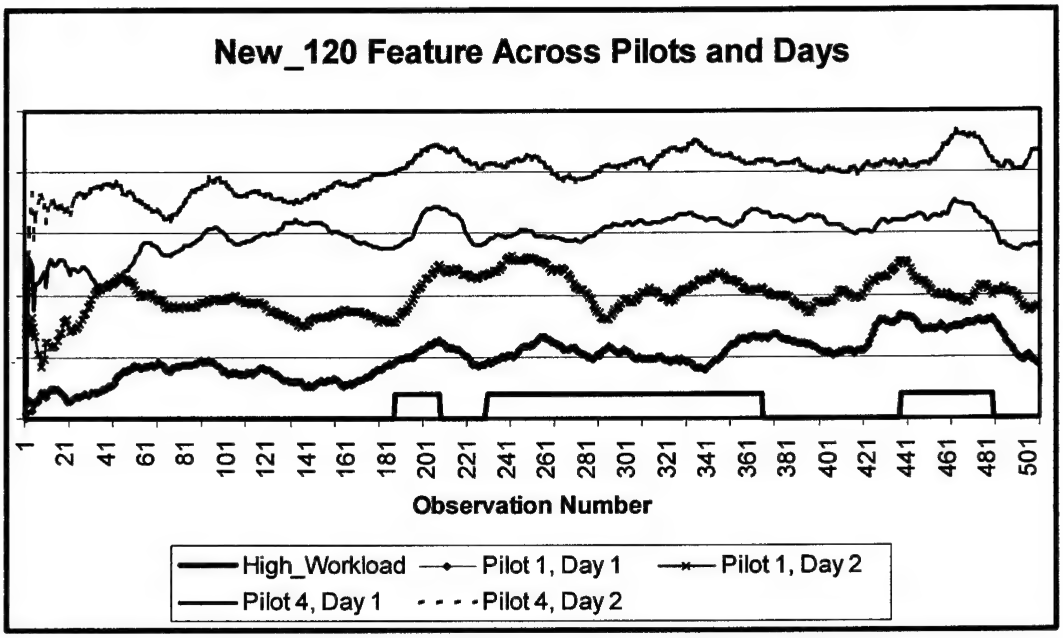


Figure 4-10. New_120 Feature Across Pilots and Days

4.5.1 Calibration with Original Workloads and Full Day Training Sets. New data sets are built for both pilots on both days using the original workloads and the calibration scheme defined in Section 4.5. The ANNs are trained using only the four new features from the calibration scheme: New_1, New_30, New_60, and New_120. All other network settings remain constant. Table 4-19 summarizes the key pieces of information associated with this section.

Table 4-19. Information Table For Calibrated Data and Full Day Data Sets

Type of Information	Description
Workload Type	Original Workload
Training Group Set	All flight segments
Data Calibrated?	Yes

4.5.2 Calibration with Original Workloads and Grouped Training Sets. The only modifications from Section 4.5.1 that occur in this section involve the training data sets. Instead of training the networks with the full day data sets, only Groups 1 and 2 are presented to them. The new training data sets incorporating the two groups are built following the same process discussed in Section 4.4.2. All other network settings remain constant. Table 4-20 summarizes the key pieces of information associated with this section.

Table 4-20. Information Table For Calibrated Data and Grouped Training Sets

Type of Information	Description
Workload Type	Original Workload
Training Group Set(s)	Groups 1 and 2
Data Calibrated?	Yes

4.5.3 Calibration with Modified Workloads and Grouped Training Sets. The final modifications to the data sets involve incorporating both the "high", "low", and "neither" workloads as well as the Group 1 and 2 training sets. The workloads are made identical to those discussed in Section 4.4.2, and the training sets are split into Groups 1 and 2 following the same process also addressed in Section 4.4.2. All other network settings remain constant. Table 4-21 summarizes the key pieces of information associated with this section.

Table 4-21. Information Table For Calibrated Data and Modified Workloads

Type of Information	**Description**
Workload Type	Modified with “High”, “Low”, “Neither”
Training Group Set(s)	Groups 1 and 2
Data Calibrated?	No

4.6 Chapter Summary

Chapter IV described the methodologies used to classify pilot mental workload. Different sections discussed the initial modeling efforts and feature reduction process, performing factor and exploratory factor analysis, modifications to the mental workload levels and different training groups, and a calibration scheme to improve network classification accuracy. Chapter V will review the results of the methodologies introduced in Chapter IV and conclude with a proposal for implementing the calibration scheme.

V. Analysis Results and Implementation Methodology

This chapter provides the results to the different methodologies introduced in Chapter IV for classifying pilot mental workload. The first section introduces several ways to measure network performance, followed by the second section that discusses the results to the initial modeling efforts after removing the non-salient features in each data set. The third section concludes the results from the exploratory factor analysis, and the fourth section presents the results from modifying the mental workload levels. The fifth section provides the results from the data calibration scheme, and the sixth section demonstrates the value of the calibration scheme through a validation effort. Finally, the seventh section introduces an implementation methodology and concludes with an implementation validation.

5.1 Evaluating Network Performance and Methodologies

Two different methods of measuring network performance are used in this chapter. The first method, introduced in Section 4.2, is classification accuracy (CA). CA is useful for summarizing a network's performance with categorical outputs in a single number. Due to how it is calculated, however, the CA measure implies equal costs of misclassification. In the case of determining pilot mental workload, we may be more interested in how accurately a network classifies high mental workload and less interested in how well it classifies low mental workload. If this is the case, then another network performance measure is needed.

A second performance measure for categorical outputs is a receiver operating characteristic (ROC). This measure is especially useful when one category is more important than others [27]. A ROC, unlike the CA measure, provides two network performance characteristics over a varying decision threshold [4]. The two characteristics are the probabilities of detection and false alarm, also known as the true positive (TP) and false positive (FP) rates. For our application, the threshold represents the cut-off probability for detecting a signal and varies from 0.0 to 1.0. For the CA measure, the threshold probability is 0.5 because this usually maximizes the probability of a correct classification [27]. Since the ROC relation ignores the separators between categories, the maximum value of a ROC typically occurs at a threshold value other than 0.5 [27]. The construction of a ROC curve is accomplished by piecing together the separate ROC true positive and false positive values and allows decision makers to readily visualize network performance and trade-off decisions.

To make the comparisons easier across the different methodologies, only the average CA and ROC values are presented. Each average is based on 12 values, and never includes the results from the same pilot and day combination used to train the network. For instance, assume a network is trained using the data from Pilot 1 on day 1. A projection of this network is then made using the data sets for Pilot 1 on day 2, Pilot 4 on day 1, and Pilot 4 on day 2. No projection is run on Pilot 1 on day 1 since this is the same pilot and day combination used to train the network. Another network is then trained using the data from Pilot 1 on day 2, and projections are made for the three other pilot and day combinations: Pilot 1 on day 1, Pilot 4 on day 1, and Pilot 4 on day 2. This process is repeated two more times using the data from Pilot 4 on day 1 and Pilot 4 on day 2 to train the networks, and data sets from the other three pilot and day combinations are projected through these two networks. The result is 12 projections, which when averaged together become one CA or ROC value. Table 5-1 shows the calculation for a single average CA or ROC value using notional data.

Table 5-1. Calculation for Average CA and ROC Value

		Projection Data Set			
		Pilot 1, Day 1	Pilot 1, Day 2	Pilot 4, Day 1	Pilot 4, Day 2
Training Data Set	Pilot 1, Day 1	▨	CA = 66% TP = .6 FP = .2	CA = 53% TP = .7 FP = .3	CA = 57% TP = .8 FP = .4
	Pilot 1, Day 2	CA = 65% TP = .8 FP = .5	▨	CA = 55% TP = .7 FP = .4	CA = 48% TP = .5 FP = .2
	Pilot 4, Day 1	CA = 60% TP = .6 FP = .3	CA = 64% TP = .7 FP = .3	▨	CA = 73% TP = .6 FP = .2
	Pilot 4, Day 2	CA = 46% TP = .5 FP = .3	CA = 48% TP = .7 FP = .4	CA = 68% TP = .8 FP = .5	▨

Average CA Value		58.58%
Average ROC Value	True Positive	0.667
	False Positive	0.333

The gray areas in the figure represent the same pilot and day combinations used for training the networks, meaning that these CA, TP, and FP values are not included in the averages. For each methodology, the average CA value is only calculated once with the cut-off threshold set at 0.5. The average TP and FP values, as mentioned earlier, are calculated 101 times to build each ROC curve as the threshold moves from 0.0 through 1.0. To simplify comparisons across the different methodologies, the same information tables from Chapter IV that identify key methodology information will precede the average results. The only modification to these tables is the addition of the average CA.

5.2 Initial Modeling Results

Following the removal of non-salient features from the different data sets and training of the ANNs, the performance measures discussed in Section 5.1 are calculated. The results of this section are consistent with those found in research by East [10].

5.2.1 SNR Saliency Screening on Individual Day Data Sets. The results from this section establish a baseline against which the other methodologies will be compared. Accordingly, these results are referred to as "baseline". The key methodology information and average CA is shown in Table 5-2, followed by the ROC curve in Figure 5-1.

Table 5-2. Baseline Information Table Results

Type of Information	Description
Workload Type	Original Workload
Training Group Set	All flight segments
Data Calibrated?	No
Average CA	59.83%

From looking at the average CA and ROC curve, we see that networks trained on the most salient features from single day data sets do not perform well across days and pilots. In fact, the ROC curve shows that the ratio of true positive to false positive rates are almost always 1:1, meaning that the trained networks provide very little information about the actual level of mental workload across pilots and days. The small arch in the ROC curve represents the limited information these networks provide.

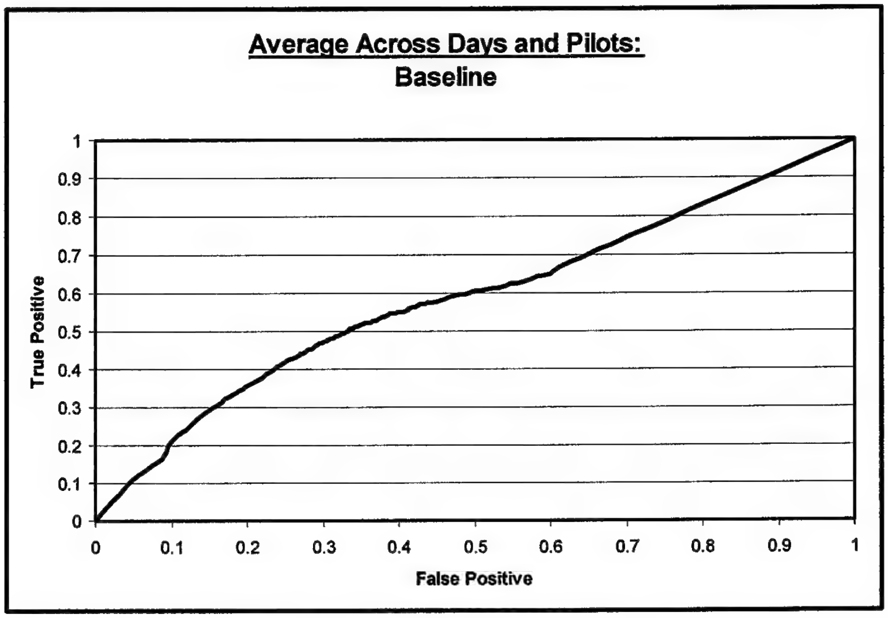


Figure 5-1. Baseline ROC Curve

5.2.2 SNR Saliency Screening on Multiple Day Data Sets. The results of the SNR saliency screening on multiple day data sets reveal that fewer features are salient for classifying Pilot 4 than Pilot 1. Furthermore, the features found most salient across the multiple day data sets are often different from those found most salient on individual day data sets, shown in Table 4-16. Possible causes for these differences include the discussion at the end of Section 4.2 concerning the randomness of the initial weights in neural networks, as well as wide variation in psychophysiological measures across days.

This variation can be a result of stress levels, sleep patterns, caffeine levels, among other causes.

Calculating network performance for the multiple day networks is secondary to the saliency screening and feature reduction results. As discussed in Section 4.5, these features are used to help determine which features to include in the calibration scheme. Nevertheless, the average CA for Pilot 1 when projected onto Pilot 4 day 1 and day 2 is 67.1%, and the average CA for Pilot 4 when projected onto Pilot 1 day 1 and day 2 is 44.33%. Given the overall average CA of 55.71%, we see that training a network over multiple day data sets does not consistently or dramatically improve our ability to accurately measure the mental workload of another pilot.

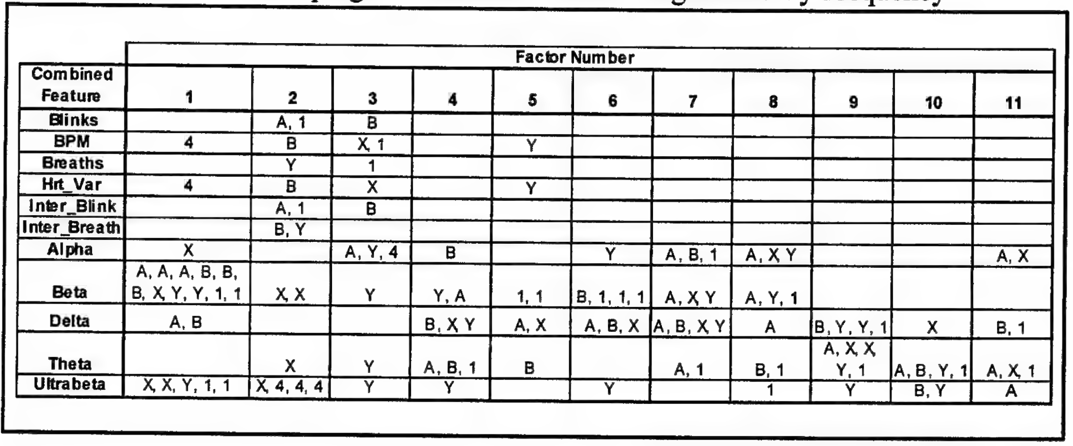
5.3 Factor Analysis

The results from conducting factor analysis on the pilot data can be found in the research by East [10], supplemented with the discussion in Section 4.3.1. In addition, most of the results from performing exploratory factor analysis on the data are already addressed in Section 4.3.2, and are used to discover the key features that show consistent patterns with changes in mental workload.

One result not fully addressed in Section 4.3.2 concerns the third grouping method of the factor data, shown in Table 4-13 and reproduced below in Table 5-3. This grouping method involves grouping the feature-to-factor assignments by frequency, meaning that the EEG node identifiers are dropped. The letter “A” indicates the results when using the data set for Pilot 1 on day 1; “B” indicates Pilot 1 on day 2; “X” indicates

Pilot 4 on day 1; "Y" indicates Pilot 4 on day 2; "1" indicates Pilot 1 over both days of data; and "4" indicates Pilot 4 over both days of data.

Table 5-3. Grouping of Feature-to-Factor Assignments By Frequency



	Factor Number										
Combined Feature	1	2	3	4	5	6	7	8	9	10	11
Blinks		A, 1	B								
BPM	4	B	X, 1		Y						
Breaths		Y	1								
Hrt_Var	4	B	X		Y						
Inter_Blink		A, 1	B								
Inter_Breath		B, Y									
Alpha	X		A, Y, 4	B		Y	A, B, 1	A, X Y			A, X
Beta	A, A, A, B, B, B, X, Y, Y, 1, 1	X, X	Y	Y, A	1, 1	B, 1, 1, 1	A, X, Y	A, Y, 1			
Delta	A, B			B, X Y	A, X	A, B, X	A, B, X Y	A	B, Y, Y, 1	X	B, 1
Theta		X	Y	A, B, 1	B		A, 1	B, 1	A, X, X Y, 1	A, B, Y, 1	A, X, 1
Ultrabeta	X, X, Y, 1, 1	X, 4, 4, 4	Y	Y		Y		1	Y	B, Y	A

Previous brain research indicates that the effects of task difficulty are mainly visible in the alpha and theta frequency bands [32]. Since the goal of this research is to accurately identify high mental workload, we hope that the most salient features across the pilots and days include many EEG nodes associated with these two frequencies. Furthermore, we also hope that when grouping the assignments by frequency and factor, we end up with the alpha and theta frequencies being associated most often with a small number of factors indicating common variation among these frequencies. Table 5-3 shows that Pilot 1 has a concentration for the alpha frequency in factor 7, whereas Pilot 4 has a concentration in factor 8. Concentrations for the theta frequency occur in factor 4 for Pilot 1, and in factor 9 for Pilot 4. Since the first few factors in factor analysis represent a proportionally larger share of the total variation in the data sets, it appears that the features associated with mental workload are not a dominant source of variation. Other frequencies, such as the beta frequency, are identified as explaining a larger portion of the total variation than the alpha and theta frequencies. This means that the first few factors in Table 5-3 might essentially represent noise when assessed as features in this mental workload classification problem, and partially explains why ANNs have such difficulty predicting pilot mental workload.

5.3.1 Network Training Results Using Key Features On Individual Data Sets. The discovery of four features that vary with changes in mental workload led to the decision to train ANNs using only these features. The four features are heart bpm, heart variability, number of blinks, and interblink. If these four features vary consistently across pilots and days, then the ANNs should learn these patterns and improve their ability to classify mental workload. Table 5-4 shows the important information associated with this network training, and the ROC curve in Figure 5-2 compares the result of training ANNs with only these four features to the baseline.

Table 5-4. Information Table Results For 4 Key Variables

Type of Information	**Description**
Workload Type	Original Workload
Training Group Set	All flight segments
Data Calibrated?	No
Average CA	58.22%

The ROC curve reveals that using just these four features actually improves the predicting capabilities by a small amount, with most of the improvement falling in the upper range of the curve. The average CA is comparable to the baseline CA. With these results, we conclude that while the performances are similar, the networks trained using only these four features are preferable to the baseline networks due to the dramatic reduction in the number of features. The baseline networks had an average of 33.75 features, whereas these networks only had 4 features.

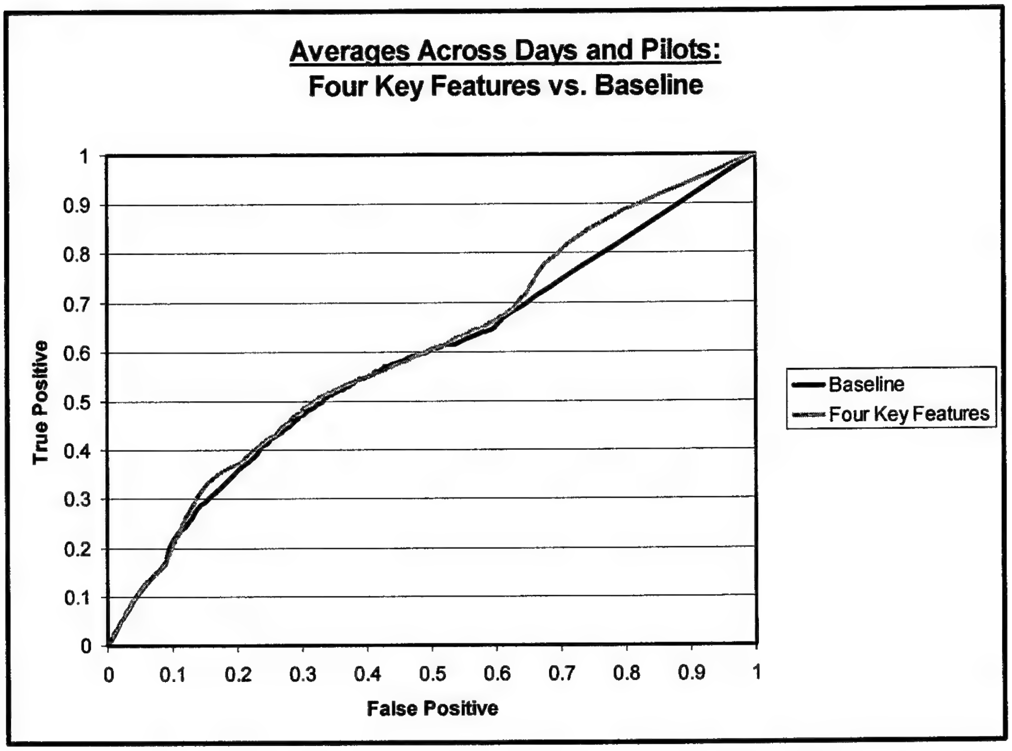


Figure 5-2. ROC Curve of Four Key Features vs. Baseline

5.4 Modified Workload Training Results

Different workload configurations are presented in Section 4.4 to challenge two assumptions used in this research effort. The first involves the assumed instantaneousness of the transitions between varying levels of workload, and the second assumption concerns how accurately the flight segments are classified by workload level. The results presented in this section challenge these assumptions by quantifying the effects of relaxing the assumptions.

5.4.1 Results From Workload Staying "High" Once Threshold Crossed. By modifying the workload levels to stay "high" once the low/high threshold is crossed, we are testing the assumption that the transitions between varying levels of workload are instantaneous. This is the situation we are trying to address: a pilot has recently finished a flight segment classified as high mental workload, and is now flying in a flight segment classified as low mental workload. Despite the lower workload in the current flight segment, does the mental workload of the pilot actually decrease, or do factors such as anticipation of approaching difficult maneuvers keep the pilot at an elevated level of mental workload? If mental workload does not actually decrease during flight segments classified as lower workload, then the modifications made to the workloads in this section should result in networks with higher classification accuracies than those in the baseline. If mental workload does decrease during flight segments classified as lower workload, then these workload level changes should cause the average CA to drop. Table 5-5 shows the key information for this section, as well as the average CA result.

This table reveals that the average CA drops approximately 1% compared to the baseline average CA of 59.83%. This means that we have no evidence to dispute the assumption that the transitions between varying levels of workload are instantaneous.

There is no need to build a ROC curve for this section since the workload modification did not result in improvement over the baseline CA.

Table 5-5. Information Table Results For High-Once-High Method

Type of Information	Description
Workload Type	Modified with high-once-high
Training Group Set	All flight segments
Data Calibrated?	No
Average CA	58.84%

5.4.2 Results From Workload Broken Into "High", "Low", and "Neither". By modifying the workload levels to include a "neither" category, we are testing the assumption that the workload levels in the flight segments are accurately classified. We are trying to determine if the flight segments classified as high mental workload are all equally high mental workload, and if the low mental workload flight segments are all equally low mental workload. Two changes are made to evaluate this assumption: the addition of the "neither" workload level indifference zone that separates high from low mental workload, and the use of training groups. The key difference between the "neither" workload level and the "medium" workload level from the original flight experiment lies in calculating the classification accuracy. With the "neither" workload level, network predictions of "neither" and low workload both count as correct predictions if the actual workload is either of these two workloads. In other words, we do not penalize the network for misclassifying within these two workload states.

If the flight segments are all accurately and equally classified, then we would expect the addition of the "neither" category to result in a drop of the average CA compared to the baseline. If the flight segments are not all accurately and equally classified, then the addition of the "neither" category should result in a significantly higher average CA than the baseline. Table 5-6 shows the key information and CA results for this section, followed by the ROC curves in Figure 5-3.

Table 5-6. Information Table Results For "High", "Low", and "Neither" Method

Type of Information	**Description**
Workload Type	Modified with "High", "Low", "Neither"
Training Group Set(s)	Groups 1 & 2
Data Calibrated?	No
Average CA	Group 1: 50.06%, Group 2: 56.36%

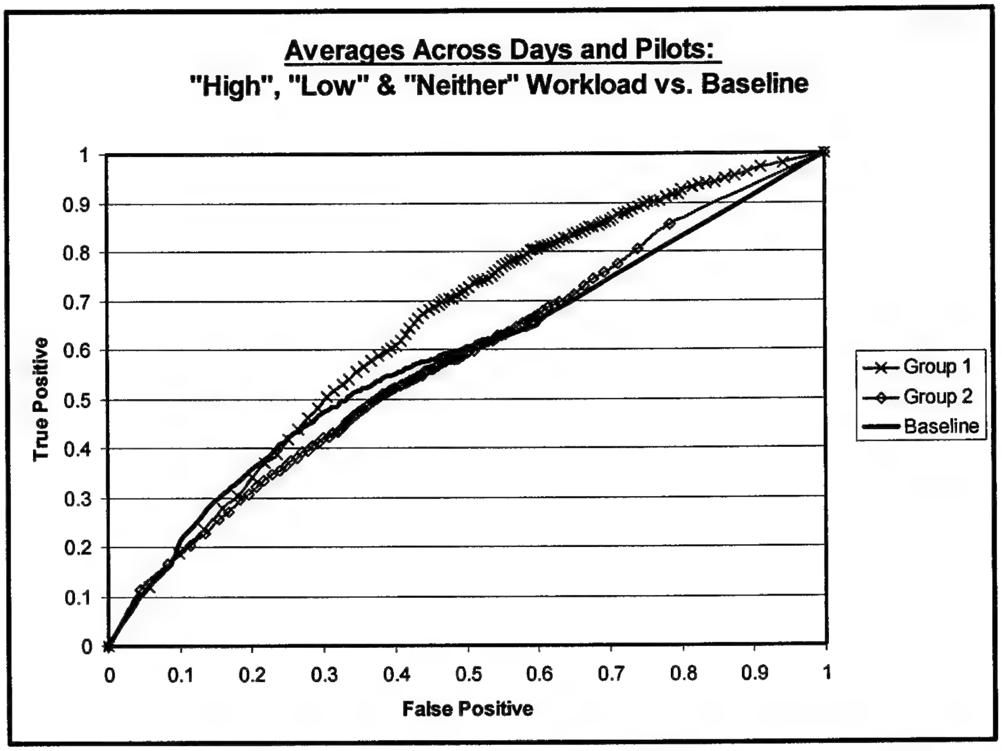


Figure 5-3. ROC Curve for "High", "Low", and "Neither" Workload Method

The average CA shows a drop of 9.77% over the baseline for the networks trained with Group1 flight segments, and a drop of 3.47% for the networks trained with the Group 2 flight segments. Despite the drop, the ROC curve for Group 1 trained networks shows improvement over the baseline curve across nearly the entire graph. The contradiction in results is attributed to the ANNs predicting a few more false alarms (causing the average CA to drop) while significantly increasing the true positive rates. The overall improvement with this workload modification gives us reason to doubt the assumption that the original workload levels are equally and accurately classified. There appear to be varying degrees of low and high mental workload, meaning that the workload levels associated with the flight segments shown in Figure 4.1 might be accurately portrayed. To accommodate the sliding scale between high and low workload, maybe the low/high workload threshold should be treated as region of indifference where the workload is neither high nor low instead of a distinct line that separates the two workload levels. This approach appears to permit ANNs to better separate the differences between high and low mental workload.

5.5 Data Calibration Scheme Results

This section presents the results of the different methods using the data calibration scheme introduced in Chapter IV. If the network performance measures show a significant improvement over the baseline results, then we conclude that the data calibration scheme works. If there is little difference to the baseline results, then we conclude that the data calibration scheme is not successful.

5.5.1 Results From Original Workloads and Full Day Data Sets. The data calibration scheme is first applied to networks using the original workload levels and full day data sets. The key information and average CA for this method is shown in Table 5-7. Figure 5-4 shows the ROC curve compared to the baseline.

Table 5-7. Information Table Results For Calibrated Data and Full Day Data Sets

Type of Information	Description
Workload Type	Original Workload
Training Group Set	All flight segments
Data Calibrated?	Yes
Average CA	72.02%

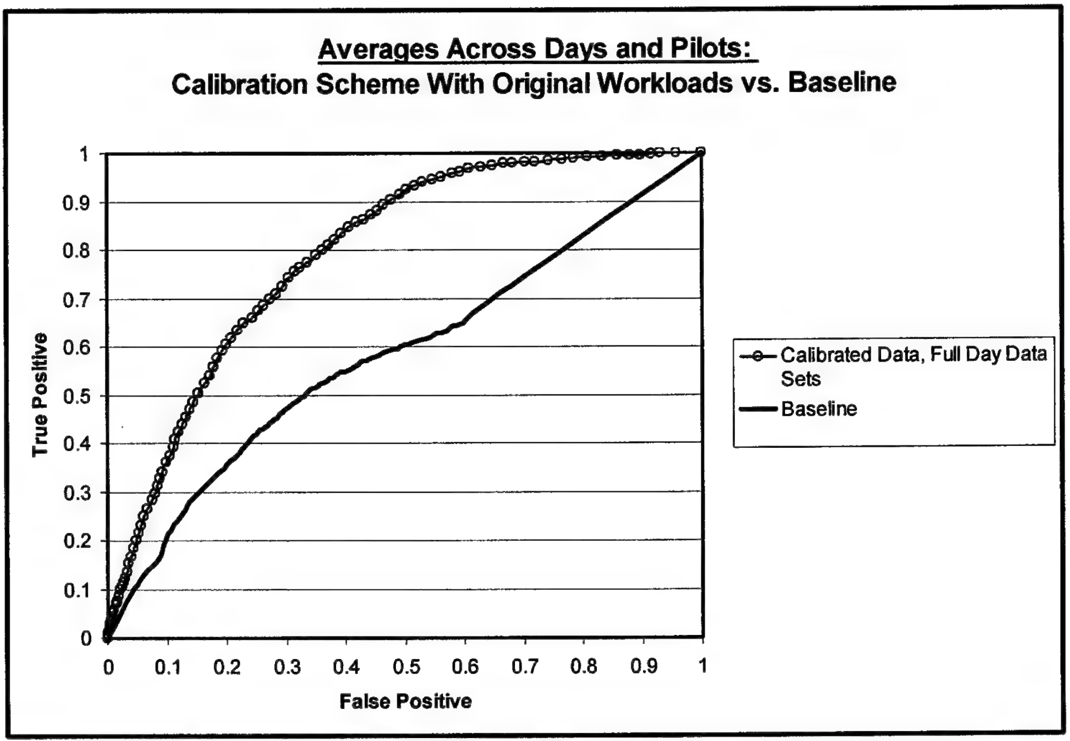


Figure 5-4. ROC Curve: Calibration, Original Workloads, and Full Day Data

The 12.19% increase in average CA over the baseline average and the dramatic improvement shown in the ROC curve clearly indicate that the calibration scheme enables ANNs to more accurately classify low and high mental workload. Chapter VI addresses why the calibration scheme improves ANN classification performance.

5.5.2 Results From Original Workloads and Use of Training Groups. The data calibration scheme is applied to the data sets with original workloads and split training groups to see if using training groups improve the network performance measures. Table 5-8 shows the key information and average CAs for this method. Figure 5-5 compares the ROC curve for this method with two other curves: the baseline ROC curve and the ROC curve from Section 5.5.1 that did not use the split training groups.

Table 5-8. Information Table Results For Calibration and Grouped Training Sets

Type of Information	**Description**
Workload Type	Original Workload
Training Group Set(s)	Groups 1 and 2
Data Calibrated?	Yes
Average CA	Group 1: 68.96%, Group 2: 70.22%

The average CA for networks trained with either Group 1 or 2 is lower than the average CA when trained on the full day data. This result is somewhat inconsistent with our expectation that the removal of flight segments around the low/high threshold would allow the ANNs to more easily distinguish the differences between low and high mental workload. Instead, it appears that the networks are able to separate the workload levels when trained on calibrated full day data, despite including the flight segments where the workload levels fall near the low/high threshold. The ROC curves support this result

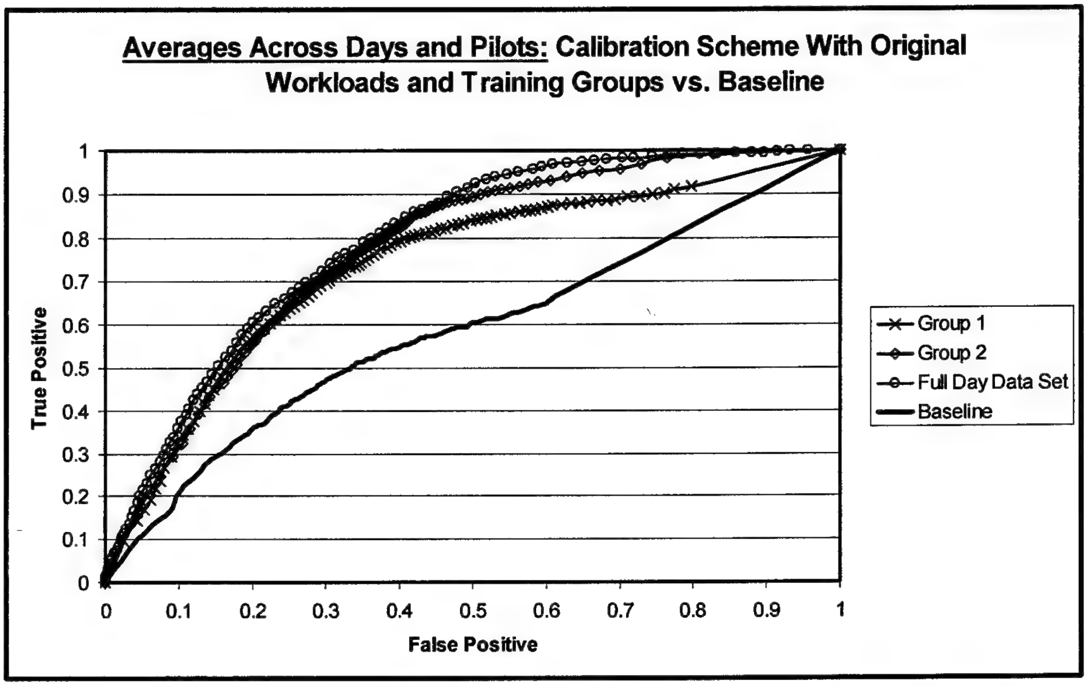


Figure 5-5. ROC Curves: Calibration, Original Workloads, and Training Groups

and show that training on a full day of data produces better results across the whole curve. Training on Group 2, which includes flight segments that fall closer to the low/high threshold, produces better results that training on Group 1 where the greatest amount of separation between workloads levels occur.

5.5.3 Results From Modified Workloads and Full Day Data Sets. The data calibration scheme is applied to the data sets with "high", "low", and "neither" workloads. From our observations in Section 5.4.2 concerning the varying degrees of high and low mental workload, we expect that the calibration scheme combined with the "neither" workload category will produce improved ROC curve performance. Table 5-9 identifies the key method information with the average CA, and Figure 5-6 shows the ROC curve compared to two other curves: the baseline ROC curve and the curve from Section 5.5.1. The only difference between the curve in this section and the curve in Section 5.5.1 is the addition of the "neither" workload group.

Table 5-9. Information Table Results For Calibration and Modified Workloads

Type of Information	Description
Workload Type	Modified with "High", "Low", "Neither"
Training Group Set	Full Day Data
Data Calibrated?	Yes
Average CA	63.01%

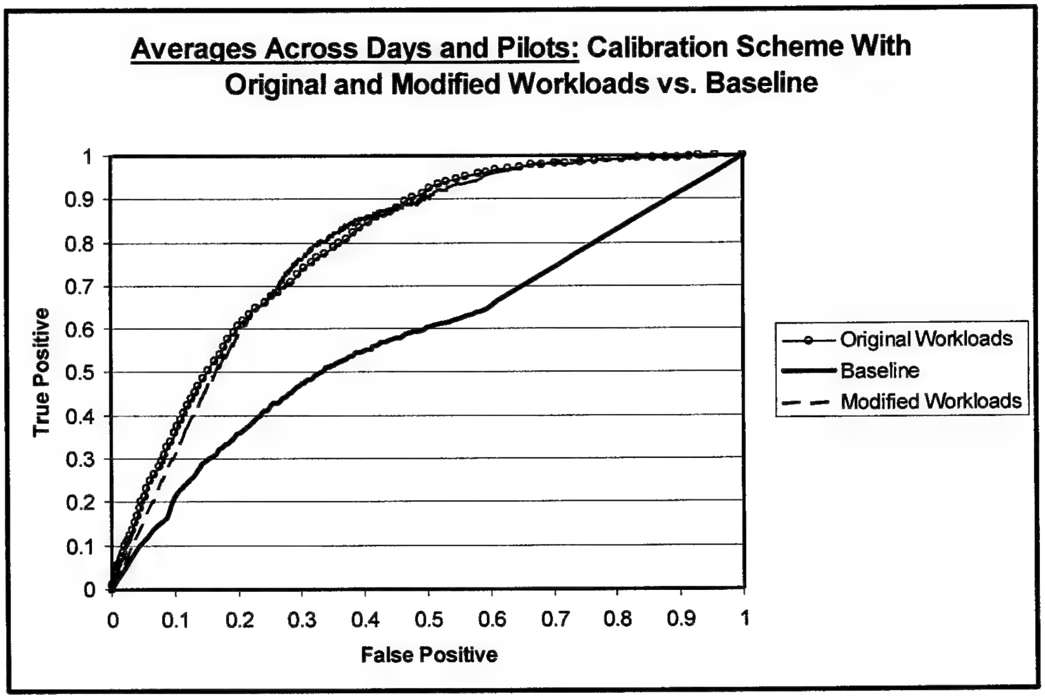


Figure 5-6. ROC Curve: Calibration, Modified Workloads, and Full Day Data Sets

The average CA decreased 9.01% over the same setup with “original” workloads, indicating that the ANNs have increased difficulties identifying differences between low and high mental workload when an indifference zone is placed between the two workload levels and the calibration scheme is used. Furthermore, Figure 5-6 shows marginal improvement between these two methods only in one portion of the curve indicating that unless the desired operating range is the middle portion of the ROC curve, using the “neither” workload category is probably unnecessary.

5.5.4 Results From Modified Workloads And Use of Training Groups. The data calibration scheme is also applied to the data sets with “high”, “low”, and “neither” workloads as well as the two training groups. The data from Section 5.5.2 indicate that networks trained using the calibration scheme and Groups 1 and 2 result in lower network performance than networks trained with calibrated full day data. If this observation holds true then we expect lower network performance in this section when compared to Section 5.5.3. Table 5-10 identifies the key method information and average CAs, and Figure 5-7 shows the ROC curves compared to two other curves: the ROC curve using full day data, and the baseline ROC curve.

Table 5-10. Information Table Results: Calibration, Modified Workload, Groups

Type of Information	**Description**
Workload Type	Modified with “High”, “Low”, “Neither”
Training Group Set(s)	Groups 1 and 2
Data Calibrated?	Yes
Average CA	Group 1: 60.85%, Group 2: 62.55%

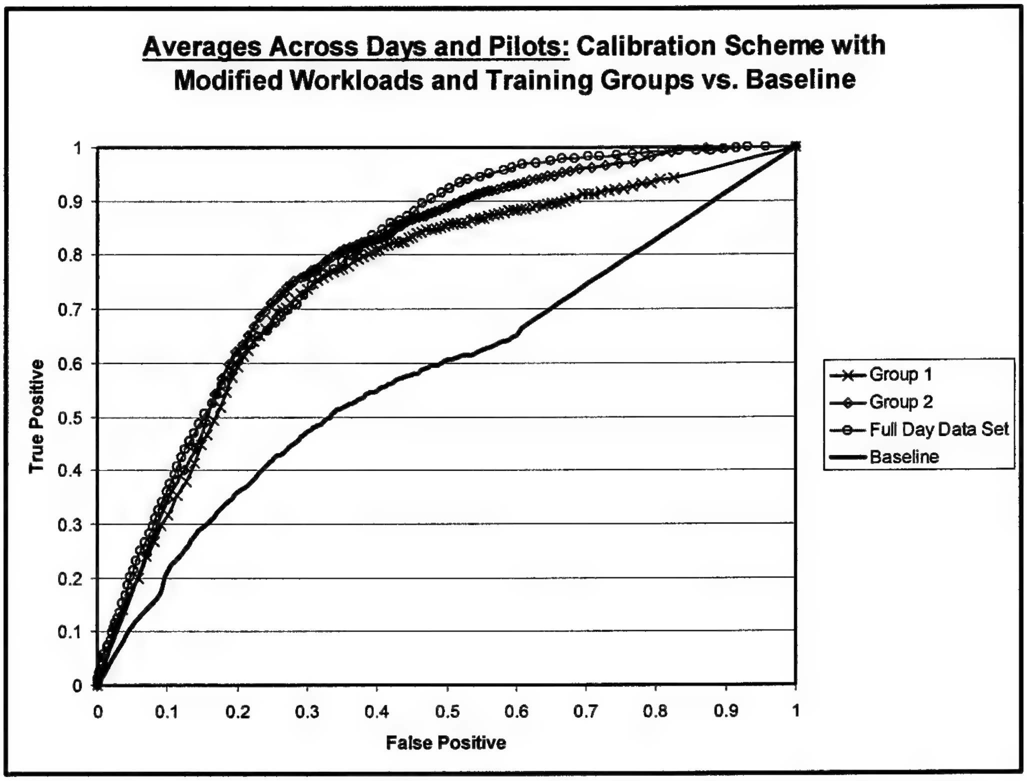


Figure 5-7. ROC Curve: Calibration, Modified Workloads, and Training Groups

The average CA values for the two training groups are lower than the average CA value when using full day data sets. In addition, Group 1 networks have lower average CAs and ROC curves than Group 2 networks. Both of these observations are consistent with the findings in Section 5.5.2. The only place where a training group performance measure surpasses the full day data set training occurs in the middle of the ROC curve. Group 2 networks have a higher ratio of true positive to false positive rates for a small portion of the ROC curve. Besides this area of the graph, however, training on all flight segments with the calibration scheme produces the highest network performance.

5.5.5 Network Training Results Using Key 4 Features Across All Data Sets. A subtle and unfair advantage is hidden in the results from Section 5.5.1. The discovery of the features that vary with changes in mental workload, which lead to the development of the particular linear combination chosen to highlight these changes, only occurred after reviewing four flights of data. The data sets used for training the ANNs in Section 5.5.1, however, consist of only one flight of data instead of four flights of data. To equal the playing field, in this section an ANN will be given a random training (and training-test) data set comprised of all four flights of data. To accomplish this, all four data sets are combined, randomly ordered, and split into training (and training-test) and validation data sets using a 60/40 ratio. If the performance from this ANN is equal to or better than the performance from Section 5.5.1, then we conclude that there is no advantage to using the calibration scheme. If the performance is lower than Section 5.5.1, then we conclude that the calibration scheme is adding value by providing additional information to the ANNs. Table 5-11 shows the important information associated with this network training, and the ROC curve in Figure 5-8 compares the result of training ANNs with these four features across all of the data sets to the calibration scheme and baseline.

Table 5-11. Information Table Results For Key Variables and Mixed Day Data

Type of Information	Description
Workload Type	Original Workload
Training Group Set	Random Data From All 4 Data Sets
Data Calibrated?	Yes
Average CA	60.67%

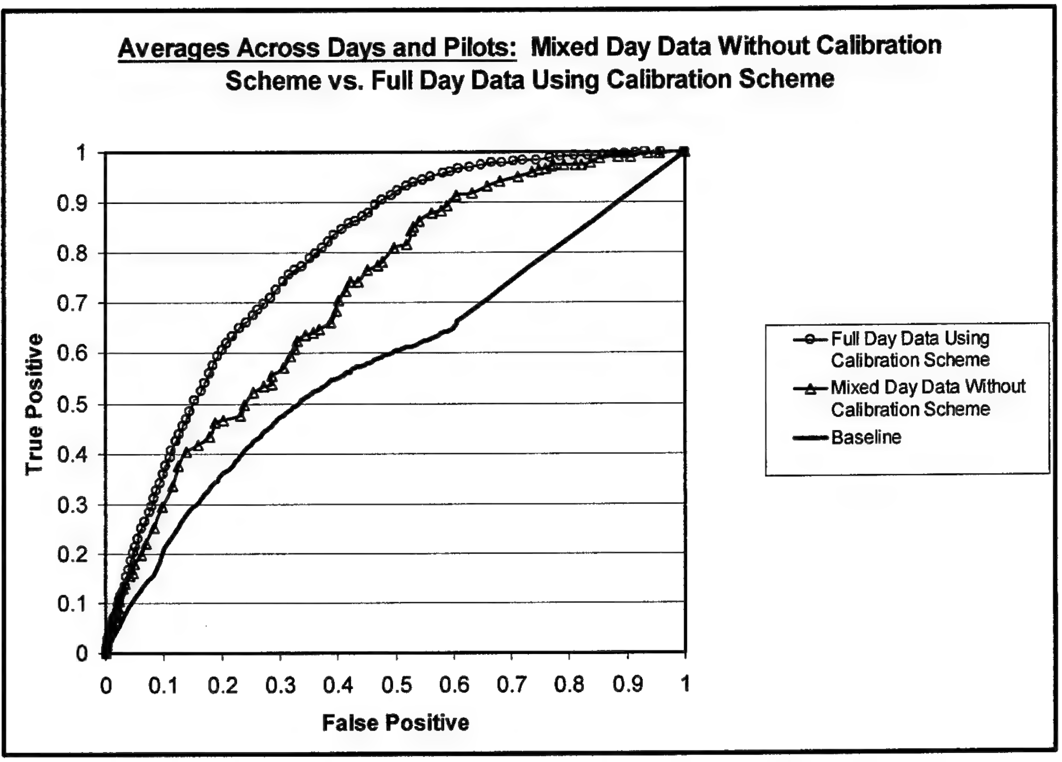


Figure 5-8. ROC Curve: Non-calibrated Mixed Day vs. Calibrated Full Day Data

The average CA for the non-calibrated mixed day data ANN is 11.35% lower than the average CA for the calibrated full day data ANNs. Furthermore, Figure 5-8 shows that the calibration scheme clearly improved network performance across the whole range of threshold values. These results indicate that the calibration scheme provides additional information to the ANNs that they cannot produce themselves. Section 6.3 addresses several reasons why this phenomenon occurs.

5.5.6. Additional Calibration Scheme Comparisons. Three additional ways to assess the benefit of using of the calibration scheme involve SNR values and rankings, measuring average network performance over the flight segments immediately following shifts in mental workload, and the variance of classification accuracies across data sets.

Table 5-12 identifies the average rank based on SNR for the features found most salient in each data set. These features are listed in Tables 4-4 through 4-7. A rank of 1.0 signifies the highest rank. Table 5-13 provides the average SNR values for these features. The tables only contain information for the four features included in the calibration scheme, and they reflect feature averages across the two pilots as well as the average per pilot. If one of the four calibration features is not listed in Tables 4-4 through 4-7, then its SNR value and rank is based on the results from the appropriate network trained with all 151 features.

Table 5-12. Average SNR Rank By Pilot Before Calibration

Source	**Heart BPM**	**Blink**	**Heart Variability**	**Interblink**
Pilot 1	1	6.5	77	2
Pilot 4	1	57	31	63.5
Average	1	31.75	54	32.75

Table 5-13. Average SNR Value By Pilot Before Calibration

Source	**Heart BPM**	**Blink**	**Heart Variability**	**Interblink**
Pilot 1	11.233	4.893	1.137	8.189
Pilot 4	18.365	2.765	2.784	0.639
Average	14.799	3.892	1.961	4.414

For comparison, Tables 5-14 and 5-15 show the same types of information except the networks are trained on 8 features: the original 4 key features and the 4 calibrated features. These tables allow us to evaluate how important the calibrated features are compared to the original features.

Table 5-14. Average SNR Rank By Pilot After Calibration

Source	**Heart BPM**	**Blink**	**Heart Variability**	**Inter blink**	**New_1**	**New_30**	**New_60**	**New_120**
Pilot 1	2.5	6.5	4.5	7.0	4.0	6.5	4.0	1.0
Pilot 4	1.0	6.0	7.5	4.5	2.5	5.0	7.0	2.5
Average	1.8	6.3	6.0	5.8	3.3	5.8	5.5	1.8

Table 5-15. Average SNR Value By Pilot After Calibration

Source	**Heart BPM**	**Blink**	**Heart Variability**	**Inter blink**	**New_1**	**New_30**	**New_60**	**New_120**
Pilot 1	10.2	1.9	8.1	2.9	6.0	1.9	7.9	11.8
Pilot 4	11.8	-0.1	-1.5	3.3	6.7	1.6	0.6	8.2
Average	11.0	0.9	3.3	3.1	6.3	1.7	4.3	10.0

Tables 5-12 and 5-13 show that heart BPM is consistently the most important feature. The other three features rank well below heart BPM, with heart variability being the only exception for Pilot 1. Tables 5-14 and 5-15 show that heart BPM remains overall the most important feature after calibration, followed very closely by New_120 which ranks second overall. New_1 and New_60 rank third and fourth overall as the next two most important features. The fifth, sixth, and seventh overall ranking features are not quite as clearly identifiable due to inconsistencies in the average values and average ranks. Nevertheless, the blinks feature overall ranks eighth. The average original feature rank is 4.9 versus the average calibration feature rank of 4.1. The average original feature SNR value is 4.6 versus the average calibration feature SNR value of 5.6.

Overall, these tables show that the calibration features dominate the original features. This information helps to explain why using the four calibration features to train ANNs produces better results.

Another way of assessing the benefit of the calibration scheme involves measuring average network performance over the flight segments immediately following shifts in mental workload. Table 5-16 shows the results over three different workload shifts: low-to-high, high-to-low, and high-to-low-to-high. The performance measure is average CA.

Table 5-16. Average CA Comparison Following Workload Shifts

Workload Shift	**Baseline**	**Calibration**
Low-to-High	54.7%	72.9%
High-to-Low	60%	55.7%
High-to-Low-to-High	54.7%	46.3%
Overall Average	56.5%	58.3%

Table 5-16 shows that the baseline method produces more consistent accuracy across the different workload shifts, despite a lower overall average CA. The low average CA from the calibration method in the high-to-low-to-high workload shift is probably due to the importance of the New_120 feature and the effects of a 2-minute moving average. Probably the most important workload shift for pilots is the low-to-high shift, and in this comparison the calibration method clearly surpassed the baseline method.

An additional method of measuring the improvements using the calibration scheme over previous classifiers involves how consistently the calibration scheme performs across different pilots and over different days. This consistency can be measured by the decrease in CA variance across the different data sets. Appendix E shows the baseline results compared to the calibration scheme results for each pilot and day combination using the original workloads and full day data sets. Overall, we find the calibration scheme reduces the CA variance by more than 88% and produces CA improvements as high as 55% over the baseline when comparing individual pilot and day combinations.

5.6 Calibration Scheme Validation

The results of the different methods using the calibration scheme indicate that ANNs trained with data using the scheme are able to better predict pilot mental workload. In order to fully determine the effectiveness and robustness of the calibration scheme, a validation effort is performed. The independent data set to be used for validation purposes comes from Pilot 6 on day 2.

To establish a baseline performance level, an ANN is trained using the original workloads levels and non-calibrated full-day data. Table 5-17 shows the information table for the baseline network, along with the average CA. The performance measures for the baseline and the calibration networks are determined by averaging the results of four projections sent through the trained networks. The four data sets sent through the networks are: Pilot 1 on days 1 and 2, and Pilot 4 on days 1 and 2.

After calibrating the data following the calibration scheme, another ANN is trained and the projections are made. Table 5-18 shows the information table along with the average CA. The ROC curve comparing the baseline to the calibration method is shown in Figure 5-9.

Table 5-17. Baseline Information Table Results

Type of Information	Description
Workload Type	Original Workload
Training Group Set	All flight segments
Data Calibrated?	No
Average CA	57.31%

Table 5-18. Calibration Validation Information Table Results

Type of Information	Description
Workload Type	Original Workload
Training Group Set	All flight segments
Data Calibrated?	Yes
Average CA	71.84%

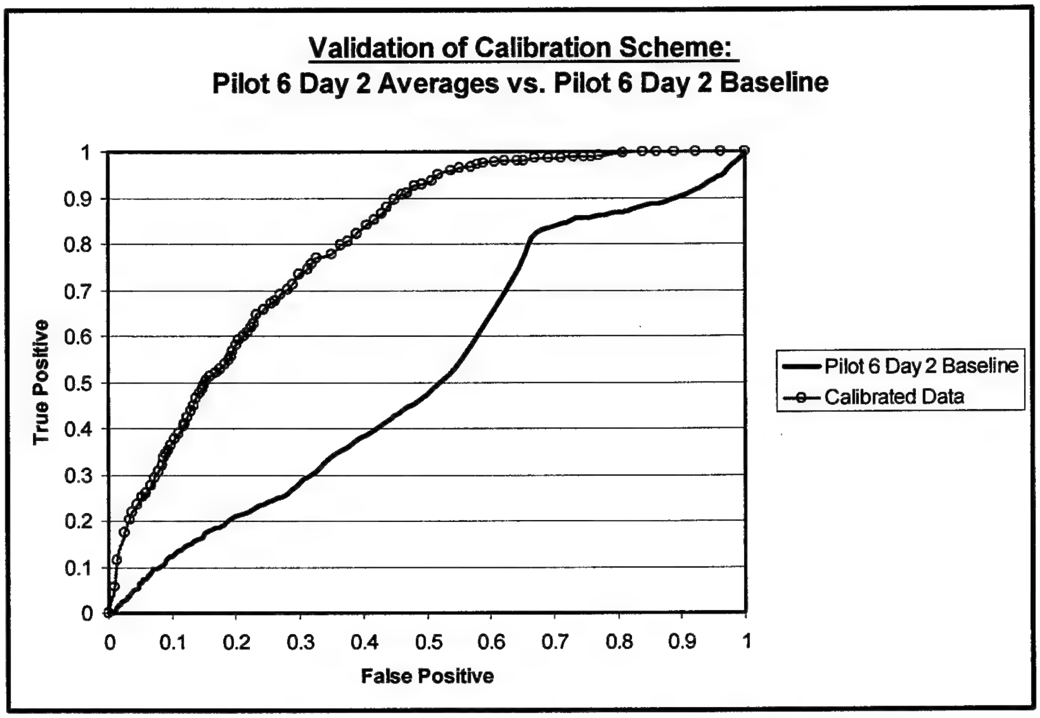


Figure 5-9. ROC Curve of Calibration Scheme Compared to Baseline

The calibration method improves average CA by 14.53% over the baseline. Furthermore, the ROC curve shows a large increase in true positive to false positive ratios across the whole curve. The performance measures in this validation effort indicate that the calibration method can be successfully applied to new data sets and result in substantially improved pilot mental workload classification.

5.7 Implementation Methodology And Validation

The calibration scheme improves network performance, however it uses the known mean and variance for each feature to produce the improved results. Since this information is unknown until the end of each flight, implementing the scheme requires some modifications. This section introduces one way to implement the calibration scheme and shows the results of a validation effort.

5.7.1 Implementation Methodology. The implementation methodology is based on constantly computing the mean and standard deviation values for each of the 4 key features (heart BPM, heart variability, number of blinks, and interblink) throughout the flight, and comparing these values to the minimum mean and standard deviation values set at the 4 minute point in the flight. Only the larger of the minimum or actual values will be used for standardizing the feature data and building the 4 calibration features. Furthermore, during the first 4 minutes of flight the pilot is assumed to be in low mental workload and the New_1 feature is set to –1.0. The other 3 calibration features, since they are moving averages of New_1 also have values of –1.0 during this period.

At 4 minutes of flight, the actual mean and standard deviation values for the 4 key features are adjusted according to the Feature Adjustment Factor Table, shown in Table 5-19. The equations for these adjustments are shown in Equations 5-1 and 5-2. These adjusted values become the minimum mean and standard deviation values for the rest of the flight. As time passes and the 4 key features are computed, they are standardized based upon the larger of the minimum mean and standard deviation values or the actual mean and standard deviation values. The New_1 calibration feature is computed using Equation 4-2, and the other 3 moving average calibration features are updated. The 4 calibration features are presented to the ANN for a prediction of current mental workload, and this process is repeated until the end of the flight. After the flight, the Feature Adjustment Factor Table should be updated to reflect the new pilot information. Alternatively, a personalized Feature Adjustment Factor Table can be built using data exclusively from one pilot. Steps 1 through 5 review the implementation process.

1. For the first 4 minutes of flight, set the New_1 feature to –1.0 to reflect the low workload state. After 4 minutes of flight, compute the actual mean and standard deviation for each of the 4 key features.

2. Find the minimum mean and standard deviation for each feature. These values are found by multiplying the actual mean and standard deviation values by the appropriate adjustment factor from the Feature Adjustment Factor Table, shown in Table 5-19. The equations to compute the minimum values are shown in Equations 5-1 and 5-2.

3. As each set of 4 key features becomes available, the continually updating mean and standard deviation for each feature is compared to the minimum values found in Step #2. If a feature mean or standard deviation value falls below the respective minimum value, then the minimum value is substituted to standardize the feature. If a feature mean or standard deviation rises above the respective minimum value, then the larger value is used to standardize the feature.

4. Using the mean and standard deviation values from Step #3, compute the New_1 calibration feature, and update the 3 moving average calibration features: New_30, New_60, and New_120. Present the 4 calibration feature values to the ANN for a prediction of current pilot mental workload.

5. Repeat Steps #3 and #4 until the end of the flight. Before the next flight begins, update the Feature Adjustment Factor Table until the values stabilize. Alternatively, update a personalized Feature Adjustment Factor Table for exclusive use by one pilot.

The Feature Adjustment Factor Table is based on data from previously flown flights. Currently, the table only reflects data from four flights: two flights by Pilot 1 and two flights by Pilot 4. Each value in the table represents the average percent difference between the overall mean (or standard deviation (SD)) for a feature and the mean (or SD) for the feature after 4 minutes of flight. To compute a minimum mean or standard deviation for feature *i*, use Equations 5-1 and 5-2 below.

$$\text{Minimum mean}_i = (\text{mean}_i \text{ after 4 minutes}) \times (1 + \text{adjustment factor}_i) \qquad (5\text{-}1)$$

$$\text{Minimum SD}_i = (\text{SD}_i \text{ after 4 minutes}) \times (1 + \text{adjustment factor}_i) \qquad (5\text{-}2)$$

Table 5-19. Feature Adjustment Factor Table

Feature	Mean Adjustment Factor	Standard Deviation Adjustment Factor
Heart Variability	-0.3707	-0.2543
Heart BPM	0.2188	0.971
Number Blinks	0.0115	0.0599
Interblink	0.1631	0.4328

5.7.2 Implementation Validation Results. To validate the implementation methodology, the data set from Pilot 6 on day 2 is used. Since information about this data set was not available during construction of the Feature Adjustment Factor Table, the two are independent of one another. The data set is processed following the implementation methodology described in Section 5.7.1. Table 5-20 shows the information table and average CA results for the implementation, and Figure 5-10 provides the ROC curve. The implementation ROC curve is compared to the baseline and the full calibration method.

Table 5-20. Calibration Implementation Information Table Results

Type of Information	Description
Workload Type	Original Workload
Training Group Set	All flight segments
Data Calibrated?	Yes, Implementation Method
Average CA	69.81%

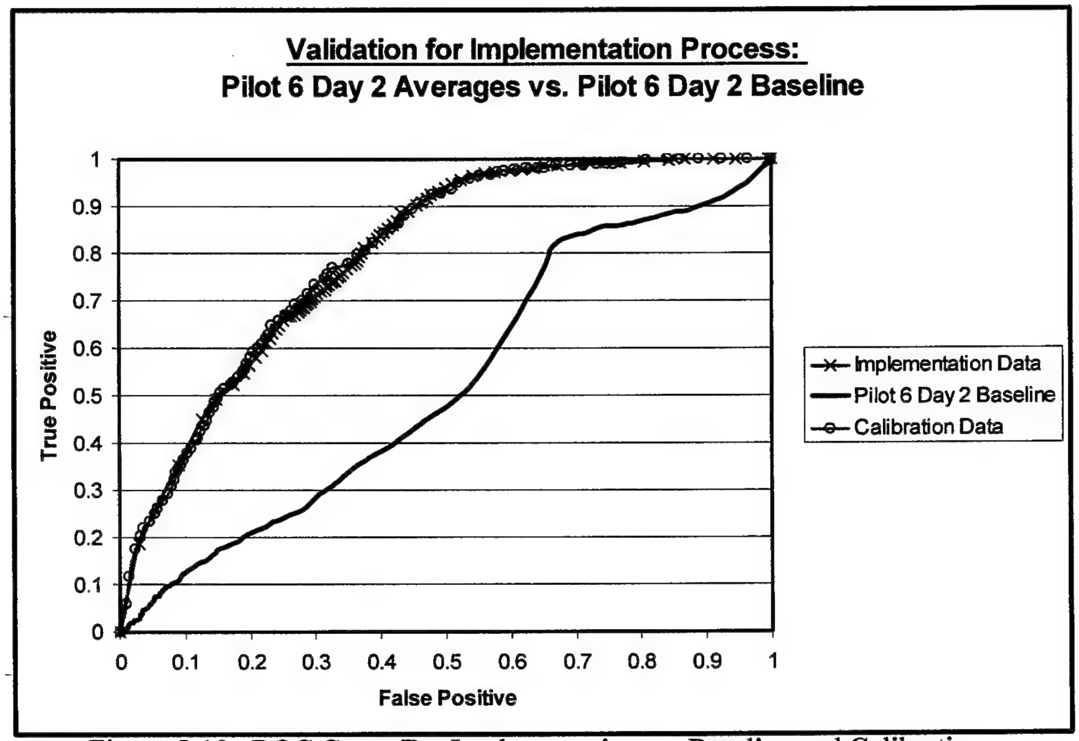


Figure 5-10. ROC Curve For Implementation vs. Baseline and Calibration

The average CA increased by 12.5% over the baseline, and the ROC curve shows the same dramatic increase over the baseline as the calibration method. Figure 5-10 also shows a comparison of the calibration and implementation methods, and the two curves are nearly identical. The middle of the graph shows an area where the full calibration method is better than the implementation method, but the improvement is small. These performance measures indicate that the implementation methodology is robust and accurately reproduces the full calibration benefits.

5.8 Chapter Summary

This chapter identified the results from the different methodologies introduced in Chapter IV for classifying pilot mental workload. Each methodology is compared to the initial modeling results, or baseline, based upon several network performance measures. Following initial success, the calibration scheme is applied to an independent data set for validation purposes, and an implementation methodology is introduced and tested. Chapter VI will take the results from this chapter and provide several conclusions about what this research has discovered. In addition to conclusions, Chapter VI will also address several recommendations for follow-on research.

VI. Conclusions and Recommendations

This chapter summarizes the results of our research effort. In particular, the research assumptions and challenges are summarized in the first section, followed by a summary of the factor analysis in the second section. The third section addresses why the calibration scheme works, and the fourth section summarizes the calibration scheme results. The fifth and final section provides several recommendations for further research.

6.1 Summary of Research Assumptions and Challenges

Two assumptions challenged in this research involve the assumptions of instantaneous transitions between varying levels of workload and perfectly specified workload levels for the flight segments. Each of these assumptions and the research findings are addressed below.

The results from Section 5.4.1 show that the "high-once-high" workload modification results in an average CA lower than the original workload levels. Since no improvement is found, we conclude we have no evidence to contradict this assumption. This does not imply that we believe the transitions between varying levels of workload are actually instantaneous. While we have no evidence to prove the assumption is invalid using our data, other research using data where test subjects simulate tasks similar to flying suggests otherwise. Laine found that the presentation order of workload appears to have a significant effect on the values of certain features [15]. He observed that inconsistencies in feature data were correlated to multiple periods of constant workload or changes from overload to low workload levels [15]. One might ask, “Why is there a difference between research results?” One possible reason involves the order of flight segments and the number of transitions across workload levels throughout the flight experiment. The flight path flown by pilots in the non-simulated flight experiment was carefully planned to include certain types of maneuvers and skills in a real-world environment. This resulted in limitations on the number of transitions from low-to-high and high-to-low workloads during the 44-minute flights. The simulated flight experiment, on the other hand, had greater flexibility to vary workloads more often since no real-world considerations like altitude, aircraft speed, and pitch had to be addressed. As a result, more transitions from low-to-high and high-to-low workloads could be included in the 45-minute simulations. Had more transitions been included in the real-world flight experiment, the workload modification method might have produced results identical to those found by Laine [15].

Another potential cause that might explain the different results concerning instantaneous transitions between the real-world and simulated flight experiments can be found in the workload levels presented to the test subjects. The simulated flight experiment included an overload workload state, where the workload difficulty is increased to the point that the test subject cannot complete all required tasks. The real-world flight experiment could not include a similar overload workload state, since this would involve serious safety risks. It is quite possible, therefore, that the transition times from simulated overload-to-medium or overload-to-low workload levels take longer than real-world experiment transitions since the overload workload state involved failure of the pilot to accomplish certain tasks. The highest workload levels during the real-world flight experiment never subjected pilots to failure of task accomplishment. This difference might be another cause for why the results in this research differ with those found by Laine [15].

The results from Section 5.4.2 indicate that there are different degrees of high and low workload, leading us to doubt that the workload levels are all accurately classified as either high or low workload. The original flight experiment split the flight segments into 3 workload levels (low, medium, and high), but previous research using this data combined the low and medium workload levels into a single low workload classification [10]. Our research initially indicates that a "neither" workload level should probably be reintroduced, based on the results shown in Figure 5-3. An interesting result occurs, however, when the calibration scheme is applied to the data and we compare the ROC curve based on the modified workload levels (low, high, and "neither") to the ROC curve based on the "original" workload levels, shown in Figure 5-6. We discover that the significant advantage to including the "neither" workload level shown in Figure 5-3 is greatly reduced. The only place on the ROC curve where including the "neither" workload level remains an advantage falls in the range of true positive values from 0.70 to 0.85. The rest of the ROC curve before, and after this true positive range, shows using 2 workload levels remains the better choice. The final determination whether or not to include the "neither" workload level, therefore, is based on the user's desired operating characteristics of the classifier.

6.2 Summary of Factor Analysis

The factor analysis method in this research effort differs slightly from a conventional factor analysis. Normally all of the factors are kept following the varimax rotation, however the large number of factors make factor interpretation difficult and do not reduce the dimensionality of our the problem. Given the ultimate goal of finding a one-size-fits-all classifier, we want to eliminate the factors that have no features assigned to them. This decision narrows the number of features to graph with workload level, ultimately leading to the discovery of 4 features that vary by workload level.

As pointed out in Chapter V, the two frequencies most associated with changes in mental workload are the alpha and theta frequencies. Factor analysis reveals that based upon their frequent association with higher number factors, these two frequencies represent less variation in the total data set than other frequencies. The other frequencies, therefore, might represent noise instead of valuable information. One possible way to eliminate this excess noise involves not including these frequencies in network training.

Overall, the factor analysis and subsequent exploratory factor analysis proved instrumental to the identification and development of the classification scheme.

6.3 Why the Data Calibration Scheme Works

The data calibration scheme works because it creates a new feature that is more immune to the psychophysiological variations that occur across different pilots and across days than the non-calibrated features by themselves. ANNs trained on non-calibrated data from one flight will have larger weight values associated with those features found to vary with the workload levels, and smaller weight values associated with those features that vary little with changes in the mental workload levels. Due to the large psychophysiological variations in different pilots and in the same pilots on different days, however, two situations occur. First, the magnitude of the changes in the features found to vary by mental workload level does not remain constant over time or by pilot. Second, the specific features that vary by mental workload level do not stay the same over time or by pilot. In other words, both the specific feature and the degree of changes vary from pilot to pilot, and by pilot over time. The SNR ranks and values in Tables 5-12 through 5-15 reflect this observation. This means that a network trained on non-calibrated data from Pilot 1 on either day will place the second greatest amount of weight on the interblink feature, and a sizable amount of weight on the number of blinks feature. Features that show less consistent variation with changes in mental workload rank beneath these features. Unfortunately, both the interblink and blink features do not show the same patterns in Pilot 4 as they do in Pilot 1, causing the trained network to result in low CA when projected onto Pilot 4 data. The SNR ranks associated with these features for Pilot 4 clearly reflect this problem. Pilot 4 has average SNR ranks for the interblink and blink features of 63.5 and 57.0, respectively. In other words, networks trained using the non-calibrated feature data on only one pilot and day stand little chance of accurately classifying mental workload across different pilots, and only a slightly better chance of accurately classifying mental workload across days for the same pilot.

The calibration scheme reduces the impacts of the psychophysiological variations that occur across different pilots and over different days. The reason this is first approached in an observation made in Section 4-3 and shown in Figures D-1 through D-

4. If one or more of the four features included in the calibration scheme are not significant to a particular pilot on a certain day, then those features basically represent small amounts of noise. Their inclusion in the linear combination results in the addition of this noise. Before a network is trained, however, the neural network software standardizes the data, thus mitigating the effect of insignificant features. As a result, the linear combination calibration scheme allows the significant features to provide valuable mental workload information to the network, and makes the insignificant features only increase the amount of noise.

Continuing the example from above using a network trained on Pilot 1 on either day, the calibration scheme adds the normalized contributions from the interblink feature, subtracts the contribution from the blink feature, adds the contribution from the heart BPM feature, and subtracts the contribution from the heart variability feature. The heart variability feature, as we see in Table 5-12, is insignificant to Pilot 1 so its addition to the calibration scheme is really an addition of noise. As mentioned before, Pilot 4 does not display the same consistent patterns as Pilot 1 in the interblink and number of blinks features, but Pilot 4 does have two consistent patterns in the heart BPM and heart variability features. This results in two features added to the calibration scheme that provide information about mental workload for Pilot 4 and two features that add noise. The outcome is a new calibration feature for Pilot 4 containing useful information about mental workload, and it can be directly compared to the calibration feature developed for Pilot 1. When data from Pilot 4 is projected through the network trained on Pilot 1, the network understandably performs quite well. The large psychophysiological variations found across different pilots and over different days is no longer a stumbling block to achieving higher classification accuracy and good ROC curve performance.

Another way to understand how the calibration scheme works involves plotting the average values of the ocular and heart features during periods of low and high workload. To reduce the number of features on the graph, the number of blinks feature is subtracted from the interblink feature to develop a single ocular feature. This is the same calculation for these features used in the calibration scheme. Similarly, the heart variability feature is subtracted from the heart BPM to create a single cardiac feature. The average values for the single ocular and cardiac features during periods of low and high workload are then calculated, producing the results shown in Table 6-1. The graph of this information is shown in Figure 6-1.

Table 6-1. Average Combined Feature Values During High and Low Workload

	Pilot 1, Day 1		**Pilot 1, Day 2**		**Pilot 4, Day 1**		**Pilot 4, Day 2**	
Source	**Low**	**High**	**Low**	**High**	**Low**	**High**	**Low**	**High**
Ocular	-0.367	0.531	-0.323	0.467	-0.021	0.030	-0.014	0.020
Cardiac	-0.339	0.489	-0.250	0.362	-0.630	0.910	-0.799	1.154
Total	-0.706	1.02	-0.573	0.829	-0.651	0.940	-0.813	1.174

Table 6-1 shows how both the ocular and cardiac values are always negative during periods of low mental workload, and always positive during periods of high mental workload. Furthermore, we observe differences in average values across the two pilots and over the two days. We notice, for example, that Pilot 4 has very small absolute values in the combined ocular feature and very large absolute values in the combined cardiac feature during both low and high workloads. Pilot 1 has combined ocular and cardiac feature absolute values that are closer in value to one another but show a stronger tendency towards the ocular feature due to its larger absolute values. These two observations confirm our results found in the exploratory factor analysis and in the SNR comparisons.

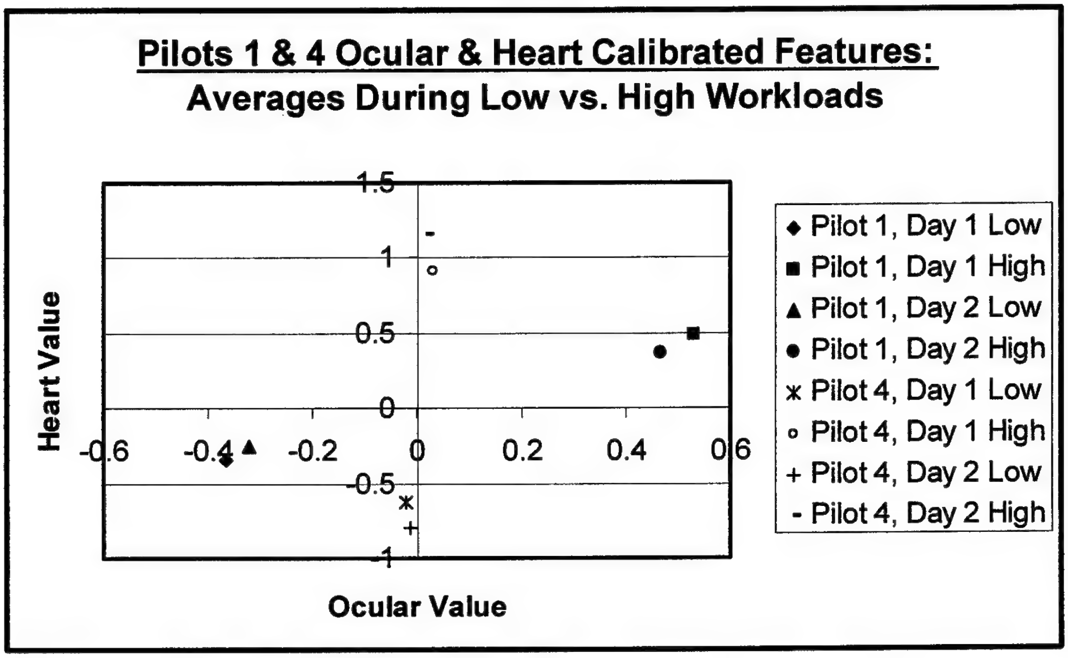


Figure 6-1. Average Combined Feature Values During High and Low Workload

Figure 6-1 visually shows the same information identified in Table 6-1. The calibration scheme clearly results in average negative values during periods of low workload and average positive values during periods of high workload, despite the differences in which features add value to the linear equation. For example, the figure shows that ocular features reflect the mental workload level for Pilot 1, where as the cardiac features reflect the mental workload level for Pilot 4. These observations are consistent with our previous findings.

6.4. Summary of Calibration Scheme Results

Our research indicates the calibration scheme dramatically improves our ability to accurately predict pilot mental workload. The validation data set shows the CA using the calibration scheme increases by more than 14% over the baseline, which is more than a 25% increase. The various ROC curves indicate even greater improvement with the calibration scheme over the baseline. Table 6-2 shows the average true positive rates for three calibration method modifications compared to the baseline when the false positive rate is set at 0.33. The table also shows the average percent of improvement over the baseline for these true positive rates.

Table 6-2. Calibration Improvement Over Baseline With FP Rate Set At 0.33

Description	**Average TP Rate**	**% Improvement Over Baseline**
Baseline with "original" workloads	0.497	-
Calibration with "original" workloads	0.774	56%
Calibration with "high, low, neither" workloads	0.800	61%
Calibration with "high, low, neither" workloads and Group 2 training	0.791	59%

Table 6-2 clearly shows how much of an improvement the calibration method makes over the baseline. In all three cases, the true positive rates improve over 55%. Furthermore, our validation effort results indicate that the calibration scheme is robust, and the implementation method results identify that the calibration scheme can be successfully implemented without any significant loss of predictive capabilities.

Through exploratory factor analysis, the reevaluation of the dimensions of the problem lead us to the insight that the feature space varies by pilot and day. While artificial neural networks appear unable to find this feature space by themselves, our calibration scheme exploits the new feature space and allows us to accurately discriminate between high and low mental workload. We achieve classification accuracy improvements over previous classifiers exceeding 55% while using 88% fewer features and reducing the classification accuracy variance by over 88%. Without the need for EEG data, the calibration scheme also reduces the raw data collection requirements by 99.75%, making data collection immensely easier to manage and dramatically reduces computational processing requirements. Along with the validated implementation method, the calibration scheme completely dominates all other classifiers over their entire operating curves and significantly simplifies the entire classification process. This makes the calibration scheme and implementation method far more practical than any previous classifier and classification method. Finally, the identification of the new feature space also opens new doors for further improvements in classification accuracies.

The calibration scheme produces a single classifier developed from only one flight that can be used to accurately predict pilot mental workload for different pilots over different days. The psychophysiological variations within and across individuals preventing previous methods from attaining high classification accuracy appear to no longer be a major hurdle.

6.5 Recommendations

Several opportunities exist for further research on calibrating pilot mental workload. The first opportunity involves exploring calibration schemes other than the linear combination presented in this research. Examples include calibration schemes containing interaction terms and non-linear functions. The second opportunity applies optimization techniques for improving the weighting of the features within the calibration scheme to optimally highlight the changes in mental workload level. Provided the predictive power and operating characteristics of the calibration scheme meets warfighter needs, the third opportunity includes moving the calibration and implementation schemes towards additional testing and future system development.

Appendix A. Microsoft Excel Feature Preprocessing Code

A.1. Preprocessing the Physiological Features

The following code preprocesses the feature data described in Chapter III. It will process two flights of data for one pilot, and it is intended for placement in separate macros where it references cells on one spreadsheet that identify certain pieces of information. This information includes: number of files to process per feature, the name and directory of the processor file storing these macros, the directory locations for data file retrieval (where the data files are located for processing one pilot over two flights), the data file prefixes, and the workload levels per flight segment.

A.1.1. Main Program Code. The main program code builds the processed data file for each flight, and then calls the other subroutine macros to process each individual feature. The main program code repeats twice to process the two flights per pilot, however only the physiological features are preprocessed by the main program code. The preprocessing time for two flights of data in this research takes approximately 2 minutes.

```
Sub Build_File_Macro()
'
' Build_File_Macro Macro
' Macro recorded 10/27/2000 by Capt Jeremy Noel
'
  Dim NumberOfFiles As Integer      ' Total number of data files'
  Dim j As Integer                  'Just a counter'
  Dim i As Integer                  'Just a counter'
  Dim Difficulty(1 To 100) As Double  'Array that holds the respective difficulty levels per flight segment
  Dim FileName2(1 To 5) As String     'Array that holds the names of the files to process
  Dim TxtMsg As String
```

```
   Dim FileNameDefault As String
   Dim RowCount As Integer
   Dim z As Integer                'Just a counter'

   Dim TxtTitle As String
   Dim Director(1 To 5) As String     'Array that holds the different data file locations
   Dim DirectorSave As String         'The directory to save the new processed data file into
   Dim Bookname(1 To 5) As String     'Array that holds the prefix names for the raw data files
   Dim FileSaveName As String         'Name of file to save, passed to subroutines
   Dim BooknameLocation As String     'Name of the bookname, passed to subroutines
   Dim DirectorLocation As String     'Name of the directory location, passed to subroutines
   Dim NameOfBook As String           'Temporary holders of the name of book
   Dim NameOfDirectory As String      'Temporary holders of the name of the directory
   Dim ColumnName(0 To 300) As String 'Names for the columns for EEG data
   Dim ColumnName2(0 To 300) As String 'Names for the columns for EEG data
   Dim Band(0 To 4) As String         'Stores the different names of the frequency bands
   Dim ColumnCounter As Integer       'A column counter
   Dim ProcessorFileName As String


'Get the directory to save the file to from the processor file
   ProcessorFileName = Cells(4, 3).Value
   Windows(ProcessorFileName).Activate
   DirectorSave = Range("D7")
'Get the number of files from the processor file'
   NumberOfFiles = Range("E3")
'Get the difficulty levels from the processor file'
   RowCount = 20
   For i = 1 To NumberOfFiles
      Difficulty(i) = Cells(RowCount, 3).Value
      RowCount = RowCount + 1
   Next i
 'Get the different file prefixes and directory locations for the two days of raw data
      NameOfBook = Range("c10")
      Bookname(1) = NameOfBook
      NameOfBook = Range("c11")
      Bookname(2) = NameOfBook
      NameOfDirectory = Range("D5")
      Director(1) = NameOfDirectory
      NameOfDirectory = Range("D6")
      Director(2) = NameOfDirectory


'*********** THIS IS THE MAJOR LOOP IN THE PROCESSING PROGRAM ************'
For z = 1 To 2

'Ask for a file name to save this processed data file as'
   TxtTitle = "Provide A File Name (8 Letters & No Spaces)"
   TxtMsg = "What name would you like to give this processed data file?  Make it 8 letters or less, include
no spaces, and use .xls for extension."
   FileNameDefault = "Pilot_Name_Day_Number.xls"

'

   'Build the new workbook'
   Workbooks.Add
```

```
ActiveCell.FormulaR1C1 = "Flt_Segment"
Range("B1").Select
ActiveCell.FormulaR1C1 = "Interval"
Range("C1").Select
ActiveCell.FormulaR1C1 = "Low_Workload"
Range("d1").Select
ActiveCell.FormulaR1C1 = "High_Workload"
Range("e1").Select
ActiveCell.FormulaR1C1 = "BPM"
Range("f1").Select
ActiveCell.FormulaR1C1 = "Hrt_Var"
Range("g1").Select
ActiveCell.FormulaR1C1 = "Blinks"
Range("h1").Select
ActiveCell.FormulaR1C1 = "Inter_Blink"
Range("i1").Select
ActiveCell.FormulaR1C1 = "Breaths"
Range("j1").Select
ActiveCell.FormulaR1C1 = "Inter_Breath"


'EEG data labels begin here
'These are all of the 29 EEG nodes: T8  O2  P10 PZ  FP1 C3  PO3 O1  IZ  P4  F3  T7  OZ  FP2 F8  P9  P3  P8  C5  CZ  FZ  F4  C6  F7  P7  FC2 PO4 FC1 C4
ColumnName(0) = "T8"
ColumnName(1) = "O2"
ColumnName(2) = "P10"
ColumnName(3) = "PZ"
ColumnName(4) = "FP1"
ColumnName(5) = "C3"
ColumnName(6) = "PO3"
ColumnName(7) = "O1"
ColumnName(8) = "IZ"
ColumnName(9) = "P4"
ColumnName(10) = "F3"
ColumnName(11) = "T7"
ColumnName(12) = "OZ"
ColumnName(13) = "FP2"
ColumnName(14) = "F8"
ColumnName(15) = "P9"
ColumnName(16) = "P3"
ColumnName(17) = "P8"
ColumnName(18) = "C5"
ColumnName(19) = "CZ"
ColumnName(20) = "FZ"
ColumnName(21) = "F4"
ColumnName(22) = "C6"
ColumnName(23) = "F7"
ColumnName(24) = "P7"
ColumnName(25) = "FC2"
ColumnName(26) = "PO4"
ColumnName(27) = "FC1"
ColumnName(28) = "C4"
'EEG Frequency bands: delta, theta, alpha, beta, ultrabeta (ubeta)
Band(0) = "delta"
```

```
  Band(1) = "theta"
  Band(2) = "alpha"
  Band(3) = "beta"
  Band(4) = "ubeta"

  'Build the labels for the EEG columns and frequency bands
  ColumnCounter = 0
  For i = 0 To (29 - 1) ' Each EEG node has 5 frequency bands

    For j = 0 To 4
    ColumnName2(i) = ColumnName(i) & "_"
      Cells(1, i + ColumnCounter + 11 + j) = ColumnName2(i) & Band(j)
    Next j
    ColumnCounter = ColumnCounter + 4


  Next i


  '***********************************'


 'Save the new processed data file'
    FileSaveName = InputBox(TxtMsg, TxtTitle, FileNameDefault)
    ActiveWorkbook.SaveAs FileName:= _
    DirectorSave & FileSaveName, _
    FileFormat:=xlText, CreateBackup:=False
    NameOfBook = Bookname(z)   'This is the file prefix
    NameOfDirectory = Director(z)  'This is the directory location
  '***********************************'

  For i = 0 To (NumberOfFiles - 1)
    For j = 1 To 23    'There are 23 exemplars per file and 2 minute segment

      Cells(j + i * 23 + 1, 1) = i + 1 'This places the flight segment into the cells
      Cells(j + i * 23 + 1, 2) = j   'This places the interval per flight segment into the cells (it can be
deleted later)
      If Difficulty(i + 1) = 1 Then
        Cells(j + i * 23 + 1, 3) = 1
        Cells(j + i * 23 + 1, 4) = 0
      Else
        Cells(j + i * 23 + 1, 3) = 0
        Cells(j + i * 23 + 1, 4) = 1
      End If
    Next j

  Next i
  Call Heart_Data(NumberOfFiles, FileSaveName, NameOfBook, NameOfDirectory)
  Call Eye_Data(NumberOfFiles, FileSaveName, NameOfBook, NameOfDirectory)
  Call Breath_Data(NumberOfFiles, FileSaveName, NameOfBook, NameOfDirectory)
'  Call FFT_Data(NumberOfFiles, FileSaveName, NameOfBook, NameOfDirectory)
  ActiveWorkbook.Save

Next z

End Sub
```

A.1.2. Cardiac Preprocessing Code. This code preprocesses the cardiac feature data. It is called from the main program code.

```
Option Explicit
Sub Heart_Data(NumberOfFiles As Integer, FileName2 As String, Bookname As String, Director As
String)

' Heart Macro
' Macro recorded 10/27/2000 by Capt Jeremy Noel

  Dim RowCount As Integer            ' Row number of .HRT file'
  Dim RowValue As Integer            ' Row Value of .HRT file'
  Dim Interval As Integer            ' Number of total Intervals from all files'
  Dim Filenumber As Integer          ' Current number of file being imported and processed'
  Dim RowTally As Integer            ' Number of beats in interval
  Dim IntervalTime As Integer        ' Current time elapsed for current interval'
  Dim RunLength As Double            ' Length of interval in milliseconds'
  Dim File As String                 ' File and directory of current file'
  Dim Number As String               ' String number extention of .HRT file
  Dim RowsInInterval As Double            ' Establishes how many rows are included in the current time
                                            window interval'
  Dim DataArray(1 To 5000) As Double 'The array that gathers each row value as it is read from the file'
  Dim i As Integer                   'Just a counter'
  Dim StarterRow As Double           'The row you started from after the last interval
  Dim SumTotal As Double             'The sum of the data values within an interval'
  Dim AverageValue(1 To 1000) As Double  'The array that holds the average values per interval
  Dim AverageValue2(1 To 1000) As Double  'The array that holds the final average values'
  Dim Slope(1 To 1000) As Double        'The array that holds the absolute value of the slope for each
                                          interval
  Dim slope2(1 To 1000) As Double    'The array that holds the final slope values'
  Dim a As Double                       'These letters, a through f, are used to calcuate the (x'x)^-1*x'y
                                        values to get the slope

  Dim b As Double
  Dim c As Double
  Dim d As Double
  Dim e As Double
  Dim f As Double
  Dim MainCounter As Double          'This keeps track of which file is being read for properly spacing the
                                      different arrays
   Dim BookNameOriginal As String      'This is the original bookname passed into the file

  BookNameOriginal = Bookname
  RunLength = 10000
  MainCounter = 0
  ' Loop runs file for all files up to variable NumberOfFiles'
  ' Adds a 0 to a single digit number '
  For Filenumber = 1 To NumberOfFiles

    If (Filenumber <= 9) Then
      Number = "0" & Filenumber
    Else
      Number = Filenumber
    End If


    Bookname = Bookname & Number & ".HRT"
     File = Director & Bookname

    ' Loads file here '
    Workbooks.OpenText FileName:=File, Origin:=xlWindows, StartRow:=1, DataType:=xlDelimited,
TextQualifier:=xlDoubleQuote, ConsecutiveDelimiter:=True, Tab:=True, Semicolon:=False,
Comma:=False, Space:=True, Other:=False, FieldInfo:=Array(Array(1, 1), Array(2, 1), Array(3, 1))

    '******** Read in the data file *****************8
    i = 1
    RowCount = 7
    While Not (Cells(RowCount, 2) = "-----------")
      RowValue = Cells(RowCount, 2).Value
      DataArray(i) = RowValue
      i = i + 1
      RowCount = RowCount + 1
    Wend

    '******** Process the odd exemplars: 1, 3, 5, ..., 23 ***********'
    Interval = 1
    RowCount = 7
    RowTally = 0
    StarterRow = 1
    IntervalTime = 0
    RowsInInterval = 0
    SumTotal = 0
    i = 1

    While Not (Cells(RowCount, 2) = "-----------")
      'Continue to add time until the 10 second RunLength has been exceeded'
      RowValue = Cells(RowCount, 2).Value
      IntervalTime = IntervalTime + RowValue


      If (IntervalTime < RunLength) Then  ' Determine if enough time has elapsed to build interval
        RowsInInterval = RowsInInterval + 1 'Increase the number of rows (and data values) included in
                                            the current interval'
      Else
        'Collect and add the data values that fell within the time interval
        a = 0
        b = 0
        c = 0
        d = 0
        e = 0
        f = 0
        SumTotal = 0
        For a = StarterRow To (StarterRow + RowsInInterval - 1)
          SumTotal = SumTotal + DataArray(a)
          b = b + a 'This line calculates the second position, or b, in the x'x matrix
          d = d + a * a
      f = f + a * DataArray(a)
    Next a
    'Average the data values within the time interval'
    AverageValue(Interval) = SumTotal / RowsInInterval 'This is the average interbeat value for this
                                                        time window'

    'Calculate the ordinary least squares estimator for b1, the slope
    'First build the x'x matrix: a,b,c,d stand for the 4 placeholders of the resulting 2x2 matrix
    a = RowsInInterval
    'b and d are already calculated above'
    c = b 'the second and third elements in the x'x matrix are identical
    'Calculated the inverse matrix: (x'x)^-1
    'To calculate the slope, we only need the 3rd and 4th elements of the (x'x)^-1 matrix
    c = -c / (a * d - b * c) 'The new c value is the 3rd element in the (x'x)^-1 matrix
    d = a / (a * d - b * b) 'The new d value is the 4th element in the (x'x)^-1 matrix
    'Now build the x'y matrix
    e = SumTotal 'This is the first element in the x'y matrix
    'f, the second element in the x'y matrix, is already calculated above'
    'Calculate the (x'x)^-1*x'y for the second element, the slope
    Slope(Interval) = ((c * e + d * f) * (c * e + d * f)) ^ 0.5 'We want only the absolute value of the
                                                          slope, so square it and take the square
                                                          root of the value

    'Reset the variables or prepare them for the next interval

    StarterRow = StarterRow + RowsInInterval
    Interval = Interval + 2
    RowCount = RowCount - 1 'The last row didn't make it into the last interval
    RowsInInterval = 0
     IntervalTime = 0
    'RowTally = 1
  End If

  RowCount = RowCount + 1
  i = i + 1

Wend

'******** Process the even exemplars: 2, 4, 6, ..., 22 ***********'
'Drop the first 5 seconds of the data'
RowTally = 0
IntervalTime = 0
RowsInInterval = 0
i = 1

While IntervalTime < 5000
  IntervalTime = IntervalTime + DataArray(i)
  i = i + 1
  RowsInInterval = RowsInInterval + 1
Wend

'Proceed with the normal development of the exemplars'
Interval = 2
RowCount = 7 + (RowsInInterval - 1) 'This eliminates the effect of the last loop where the sum fell
                                     above the limit
```

```
RowTally = 0
StarterRow = RowsInInterval
IntervalTime = 0
RowsInInterval = 0
SumTotal = 0
i = 1

While Not (Cells(RowCount, 2) = "----------")
  'Continue to add time until the 10 second RunLength has been exceeded'
  RowValue = Cells(RowCount, 2).Value
  IntervalTime = IntervalTime + RowValue


  If (IntervalTime < RunLength) Then  ' Determine if enough time has elapsed to build interval
    RowsInInterval = RowsInInterval + 1 'Increase the number of rows (and data values) included in
                                         the current interval'

  Else
    'Collect and add the data values that fell within the time interval
    a = 0
    b = 0
    c = 0
    d = 0
    e = 0
    f = 0
    SumTotal = 0
    For a = StarterRow To (StarterRow + RowsInInterval - 1)
      SumTotal = SumTotal + DataArray(a)
      b = b + a 'This line calculates the second position, or b, in the x'x matrix
      d = d + a * a
      f = f + a * DataArray(a)
    Next a
    'Average the data values within the time interval'
    AverageValue(Interval) = SumTotal / RowsInInterval 'This is the average interbeat value for this
                                                        time window'

    'Calculate the ordinary least squares estimator for b1, the slope
    'First build the x'x matrix: a,b,c,d stand for the 4 placeholders of the resulting 2x2 matrix
    a = RowsInInterval
    'b and d are already calculated above'
    c = b 'the second and third elements in the x'x matrix are identical
    'Calculated the inverse matrix: (x'x)^-1
    'To calculate the slope, we only need the 3rd and 4th elements of the (x'x)^-1 matrix
    c = -c / (a * d - b * c) 'The new c value is the 3rd element in the (x'x)^-1 matrix
    d = a / (a * d - b * b) 'The new d value is the 4th element in the (x'x)^-1 matrix
    'Now build the x'y matrix
    e = SumTotal 'This is the first element in the x'y matrix
    'f, the second element in the x'y matrix, is already calculated above'
    'Calculate the (x'x)^-1*x'y for the second element, the slope
    Slope(Interval) = ((c * e + d * f) * (c * e + d * f)) ^ 0.5 'We want only the absolute value of the
                                                                 slope, so square it and take the square
                                                                 root of the value

   'Reset the variables or prepare them for the next interval
    StarterRow = StarterRow + RowsInInterval
```

```
            Interval = Interval + 2
            RowCount = RowCount - 1 'The last row didn't make it into the last interval
            RowsInInterval = 0
            IntervalTime = 0
            'RowTally = 1
         End If

         RowCount = RowCount + 1
         i = i + 1

      Wend

      'Keep only the first 23 exemplars from each file
      For i = 1 To 23
         AverageValue2(i + 23 * MainCounter) = AverageValue(i)
         slope2(i + 23 * MainCounter) = Slope(i)
      Next i

      ' Close current workbook '
      Windows(Bookname).Activate
      ActiveWorkbook.Close
      MainCounter = MainCounter + 1

      Bookname = BookNameOriginal

   Next Filenumber

'**********************************************************************'
   ' Place the processed data into the processed data worksheet'

   Windows(FileName2).Activate
    For i = 1 To 23 * NumberOfFiles 'There are 23 exemplars per file
       AverageValue2(i) = 60000 * 1 / AverageValue2(i)   'To get a beats per minute value, invert the
                                                          average time between beats and multiply by
                                                          60,000
      Cells(i + 1, 5) = AverageValue2(i)
      Cells(i + 1, 6) = slope2(i)

   Next i

   ActiveWorkbook.Save

End Sub
```

A.1.3. Ocular and Respiratory Preprocessing Code. This code preprocesses the ocular feature data. With only file extension changes it also preprocesses the respiratory feature data. Both of these subroutines are called from the main program code.

```
Option Explicit
Sub Eye_Data(NumberOfFiles As Integer, FileName2 As String, Bookname As String, Director As String)

'
' Eye_Data Macro
' Macro recorded 10/28/2000 by Capt Jeremy B. Noel
'

'
  Dim RowCount As Double           ' Row number of file'
  Dim Interval As Integer          ' Number of total Intervals from all files'
  Dim Filenumber As Integer        ' Current number of file being imported and processed'
  Dim IntervalTime As Double       ' Current time elapsed for current interval'
  Dim RunLength As Double          ' Length of interval in milliseconds'
  Dim File As String               ' File and directory of current file'
  Dim Number As String             ' String number extention of  file
  Dim RowsInInterval As Double     ' Establishes how many rows are included in the current time window
                                     interval'
  Dim DataArray1(1 To 5000) As Double  'The array for odd exemplars that gathers each row value as it is
                                     read from the file'
  Dim i As Double                  'Just a counter'
  Dim StarterRow As Double         'The row you started from after the last interval
  Dim SumTotal As Double           'The sum of the data values within an interval'
  Dim AverageValue1(1 To 1000) As Double  'The array that holds the average values per interval
  Dim AverageValue2(1 To 1000) As Double  'The array that holds the final average values'
  Dim a As Double                  'Just a counter'
  Dim MainCounter As Double         'This keeps track of which file is being read for properly spacing the
                                     different arrays
  Dim NumberBlinks1(-1 To 1000) As Double   'Array that holds the number of blinks in each 10 second
                                     time interval
  Dim NumberBlinks2(-1 To 1000) As Double   'Array that holds the final number of blinks in each 10
                                     second time interval
  Dim RowValue As Double           'The value in the respective row of the file'
  Dim TimeWindow As Double          'Counter keeping track of which time window we are in
  Dim RowsInFirstFive As Double     'Set as the number of rows that fall within the first 5 seconds
  Dim BookNameOriginal As String     'This is the original bookname passed into the file


  BookNameOriginal = Bookname
  RunLength = 10000
  MainCounter = 0


  ' Loop runs file for all files up to variable NumberOfFiles'

  ' Adds a 0 to a single digit number '
For Filenumber = 1 To NumberOfFiles

    If (Filenumber <= 9) Then
      Number = "0" & Filenumber
    Else
      Number = Filenumber
    End If
    Bookname = Bookname & Number & ".BLK"
    File = Director & Bookname

    ' Loads file here '
    Workbooks.OpenText FileName:=File, Origin:=xlWindows, StartRow:=1, DataType:=xlDelimited,
TextQualifier:=xlDoubleQuote, ConsecutiveDelimiter:=True, Tab:=True, Semicolon:=False,
Comma:=False, Space:=True, Other:=False, FieldInfo:=Array(Array(1, 1), Array(2, 1), Array(3, 1))

    '******** Read in the data file *****************
    i = 1
    RowCount = 7
    While Not (Cells(RowCount, 2) = "-----------")
      RowValue = Cells(RowCount, 2).Value
      DataArray1(i) = RowValue
      i = i + 1
      RowCount = RowCount + 1
    Wend

    '******** Process the odd exemplars: 1, 3, 5, ..., 23 ***********'
    Interval = 1
    RowCount = 7
    StarterRow = 1
    IntervalTime = 0
    RowsInInterval = 0
    SumTotal = 0
    TimeWindow = 1


    While Not (Cells(RowCount, 2) = "-----------")
      'Continue to add time until the 10 second RunLength has been exceeded'
      RowValue = Cells(RowCount, 2).Value
      IntervalTime = IntervalTime + RowValue


      If (IntervalTime < RunLength * TimeWindow) Then ' Determine if enough time has elapsed to
                                                        build interval
        RowsInInterval = RowsInInterval + 1 'Increase the number of rows (and data values) included in
                                                        the current interval'
      Else
         NumberBlinks1(Interval) = RowsInInterval - StarterRow + 1 'This puts the number of blinks in
                                                        the interval into the array

        'Calculate the average time between blinks
        If NumberBlinks1(Interval) > 1 Then
          SumTotal = 0
          For a = StarterRow To RowsInInterval
            SumTotal = SumTotal + DataArray1(a)
          Next a
          StarterRow = StarterRow + NumberBlinks1(Interval)
          AverageValue1(Interval) = (SumTotal / NumberBlinks1(Interval)) / 1000
        ElseIf NumberBlinks1(Interval) = 1 Then  'Use the time between the last blink and the one blink
                                                        in the interval
          AverageValue1(Interval) = DataArray1(RowCount - 7) / 1000
          StarterRow = StarterRow + NumberBlinks1(Interval)
      ElseIf NumberBlinks1(Interval) = 0 Then  'If no blinks occur, subtract the time of the last blink
                                                          from the end of the current window
        SumTotal = 0
        a = 1
        While a < StarterRow
          SumTotal = SumTotal + DataArray1(a)
          a = a + 1
        Wend
        AverageValue1(Interval) = (RunLength * TimeWindow - SumTotal) / 1000
      End If

      'Reset the variables or prepare them for the next interval
      Interval = Interval + 2
      RowCount = 6 'Each time you want to read through the entire data set until the main condition is
                 met
      RowsInInterval = 0
      IntervalTime = 0
      TimeWindow = TimeWindow + 1

    End If

    RowCount = RowCount + 1

  Wend

'******** Process the even exemplars: 2, 4, 6, ..., 22 ***********'
'Drop the first 5 seconds of the data'

IntervalTime = 0
RowsInInterval = 0

i = 1
While IntervalTime < 5000
  IntervalTime = IntervalTime + DataArray1(i)
  RowsInInterval = RowsInInterval + 1
  i = i + 1
Wend


'Proceed with the normal development of the exemplars'
Interval = 2
RowsInFirstFive = RowsInInterval - 1    'The -1 eliminates the looping structure's extra +1 from above
RowCount = 7
StarterRow = 1
IntervalTime = 0
RowsInInterval = 0
SumTotal = 0
TimeWindow = 1


While Not (Cells(RowCount, 2) = "----------")
  'Continue to add time until the 10 second RunLength has been exceeded'
  RowValue = Cells(RowCount, 2).Value
  IntervalTime = IntervalTime + RowValue
```

```
If (IntervalTime < (RunLength * TimeWindow + 5000)) Then ' Determine if enough time has
                                    elapsed to build interval
  RowsInInterval = RowsInInterval + 1 'Increase the number of rows (and data values) included in
                                    the current interval'

Else
  NumberBlinks1(Interval) = RowsInInterval - StarterRow + 1 - RowsInFirstFive 'This puts the
                                    number of blinks in the interval into the array

  'Calculate the average time between blinks
  If NumberBlinks1(Interval) > 1 Then
    SumTotal = 0
    For a = (StarterRow + RowsInFirstFive) To RowsInInterval
      SumTotal = SumTotal + DataArray1(a)
    Next a
    AverageValue1(Interval) = (SumTotal / NumberBlinks1(Interval)) / 1000
    StarterRow = StarterRow + NumberBlinks1(Interval)
  ElseIf NumberBlinks1(Interval) = 1 Then  'Use the time between the last blink and the one blink
                                    in the interval
    AverageValue1(Interval) = DataArray1(RowCount - 7) / 1000
    StarterRow = StarterRow + NumberBlinks1(Interval)
  ElseIf NumberBlinks1(Interval) = 0 Then  'If no blinks occur, subtract the time of the last blink
                                    from the end of the current window
    SumTotal = 0
    a = 1
    While a < (StarterRow + RowsInFirstFive)
      SumTotal = SumTotal + DataArray1(a)
      a = a + 1
    Wend
    AverageValue1(Interval) = (RunLength * TimeWindow + 5000 - SumTotal) / 1000
  End If

  'Reset the variables or prepare them for the next interval
  Interval = Interval + 2
  RowCount = 6 'Each time you want to read through the entire data set until the main condition is
                                    met
  RowsInInterval = 0
  IntervalTime = 0
  TimeWindow = TimeWindow + 1

End If

RowCount = RowCount + 1

Wend

'Keep only the first 23 exemplars from each file
For i = 1 To 23
  NumberBlinks2(i + 23 * MainCounter) = NumberBlinks1(i)
  AverageValue2(i + 23 * MainCounter) = AverageValue1(i)

Next i

' Close current workbook '
```

```
    Windows(Bookname).Activate
    ActiveWorkbook.Close
    MainCounter = MainCounter + 1
    Bookname = BookNameOriginal

Next Filenumber


'*********************************************************************'
  ' Place the processed data into the processed data worksheet'
  Windows(FileName2).Activate

  For i = 1 To 23 * NumberOfFiles 'There are 23 exemplars per file
    Cells(i + 1, 7) = NumberBlinks2(i)
    Cells(i + 1, 8) = AverageValue2(i)
  Next i

  ActiveWorkbook.Save

End Sub
```

A.2. Preprocessing the EEG Features

The EEG feature preprocessing requires both Microsoft Word and Microsoft Excel. Microsoft Word is needed due to memory management issues with Microsoft Excel and Microsoft Windows. The main program runs in Word and after each 2-minute EEG file is processed, it shuts down Excel and re-opens it free the computer RAM. As a result, the main program code is placed in a Word macro and the other code in Section A.2.2 is placed in an Excel macro. Processing time for each flight of data takes approximately 22 hours since over 19.5 million FFTs are performed, sorted, and recorded per flight. This time estimate is based on a networked 850MHz computer with 512MB RAM, however automatic network and disk scanning functions periodically delayed the processing speed over this period of time.

A.2.1. Main Program Code For Placement In Microsoft Word Macro. This macro code needs to be placed in Word since the EEG preprocessing is controlled by Word, not Excel. This macro calls other macros in Excel, shown in Section A.2.2.

```
Sub Execute_FFT_Pilot1_Day1_In_excel()
'
' Execute_FFT_Pilot1_Day1_In_excel Macro
' Macro recorded 11/8/00 by ENS
'

Dim ProcessFile As Object
Dim OutputFile As Object
Dim LastExcelSheet As Object

'**********************************************************************
'ALL FILE MODIFICATIONS OCCUR RIGHT HERE
DFileName = "Pilot1_Day1.xls"
DataFileName = "c:\Capt Noel Thesis\Pilot1_Day1.xls"
MacroToRunName = "'ProcessorFile for all pilots.xls'!FFT_Pilot1_Day1"

'**********************************************************************

'NO MODIFICATIONS NECESSARY BEYOND THIS POINT IN PROGRAM
'Open the processor file in excel and reset the counters

LastExcelSheetName = "LastExcelSheet.xls"
ProcessFileName = "c:\Capt Noel Thesis\ProcessorFile for all pilots.xls"
PFileName = "ProcessorFile for all pilots.xls"
Set OutputFile = GetObject(DataFileName)
Set ProcessFile = GetObject(ProcessFileName)
Set LastExcelSheet = GetObject(LastExcelSheetName)

ProcessFile.Application.Visible = True
ProcessFile.Windows(PFileName).Visible = True
ProcessFile.Application.Cells(15, 15).Value = 1 'The original file number setting is 1
ProcessFile.Application.Cells(14, 15).Value = 0 'The original main counter value is 0
ProcessFile.Close SaveChanges:=True
LastExcelSheet.Application.Quit

'This is just a pausing statement to allow excel to fully close before being re-opened
  For j = 1 To 500000000
    a = a
  Next j

'****************************************
'The main loop starts here
For FileNumber = 1 To 22

'For speed this might not be wanted, visible=true.  Maybe just comment it out after error checking
```

```
ProcessFile.Application.Visible = True

' Need to add something about the "enable macro" default value here. ("yes" vs. "no" default)
' NOTE: IF COMPUTER ASKS FOR ENABLING MACROS WHEN OPENING FILE, CHANGE MACRO SECURITY SETTINGS TO LOW, ELIMINATING THIS PROBLEM
ProcessFile.Windows(PFileName).Visible = True
OutputFile.Windows(DFileName).Visible = True

'***** process data here ******
'Put which loop number we are currently processing into the processor spreadsheet
ProcessFile.Application.Cells(15, 15).Value = FileNumber
'call macro to process here
'ProcessFile.Application.Run "'ProcessorFile mod for eeg process.xls'!Build_File_Macro"
ProcessFile.Application.Run MacroToRunName

'****** stop processing data here ******
'Saving and closing portion below only'

OutputFile.Close SaveChanges:=True
'ProcessFile.SaveAs savenameprocess
ProcessFile.Close SaveChanges:=True
LastExcelSheet.Application.Quit

'This is just a pausing statement to allow excel to fully close before being re-opened
  For j = 1 To 500000000
    a = a
  Next j

'End of main loop here
Next FileNumber

End Sub
```

A.2.2. Code for EEG Preprocessing For Placement In Microsoft Excel. This code is currently set-up to preprocess one flight of EEG data at a time, and each flight of data for preprocessing has its own macro. To accomplish this task easily, copy the macro multiple times and only modify the key information following the variable declarations to preprocess each different flight. An alternative is to have a separate location on an Excel spreadsheet that identifies the header information to process multiple flights of data using only one Excel macro.

```
Option Explicit
Sub FFT_Pilot1_Day1()

'
' FFT_Pilot1_Day1 Macro
' Macro 11/7/2000 by Capt Jeremy Noel
'
Dim Matches As Integer     'The number of matches when re-arranging the cells for processing
Dim RowNumber As Integer 'The number of the row
Dim CellValue As Double 'The value of the cell
Dim CellValue2 As Variant 'A modified value of a cell: either a number if positively valued, or '(number)
(an added appostraphy) if negatively valued
Dim i As Double          ' Just a counter
Dim j As Double          ' Just a counter
Dim b As Double          ' Just a counter
Dim c As Double          ' Just a counter
Dim d As Double          ' Just a counter
Dim m As Double          ' Just a counter
Dim n As Double          ' Just a counter
Dim o As Double          ' Just a counter
Dim DataStart As Double    'Location where the data starts to process within each data file
Dim DataEnd As Double      'Location where the data ends to process within each data file
Dim DataStart2 As Double    'Location where the data starts to process within each data file
Dim DataEnd2 As Double      'Location where the data ends to process within each data file
Dim VEOGLocation As Integer 'The location of the data column of VEOG
Dim HEOGLocation As Integer 'The location of the data column of HEOG
Dim ColumnLabels(1 To 29) As String 'This array holds the 29 column labels
Dim ColumnName(0 To 50) As String  'Names for the columns for EEG data
Dim BookNameOriginal As String     'This is the original bookname passed into the file
Dim DeltaArray(1 To 3000) As Double 'Array to hold the processed Delta information
Dim ThetaArray(1 To 3000) As Double 'Array to hold the processed Delta information
Dim AlphaArray(1 To 3000) As Double 'Array to hold the processed Delta information
Dim BetaArray(1 To 3000) As Double 'Array to hold the processed Delta information
Dim UBetaArray(1 To 3000) As Double 'Array to hold the processed Delta information
Dim AverageD As Double            'Holds the average for the varioius frequency bands
Dim AverageT As Double            'Holds the average for the varioius frequency bands
Dim AverageA As Double            'Holds the average for the varioius frequency bands
Dim AverageB As Double            'Holds the average for the varioius frequency bands
Dim AverageU As Double            'Holds the average for the varioius frequency bands
Dim r1 As Range                'Holds a part of a range of cells
Dim r2 As Range                'Holds a part of a range of cells
Dim myMultiAreaRange As Range       'The combined ranges of cells for deleting
Dim NameOfProcessorFile As String  'This is the name of the main processor file
Dim CellCheck As Double           'This checks for all eeg nodes
Dim MainCounter As Double            'This keeps track of which file is being read for properly spacing the
                                      different arrays
Dim Filenumber As Integer         ' Current number of file being imported and processed'
Dim File As String             ' File and directory of current file'


'**** Because this takes so long to process, keep these in here instead of auto processing ******
Dim NumberOfFiles As Integer
Dim FileName2 As String
Dim Bookname As String
Dim Bookname1(1 To 22) As String
Dim Director As String
Dim e As Integer


'************************************************************
'******* MODIFY ONLY THESE LINES TO PROCESS PILOT X DAY Y ******

'*** This part processes Pilot 1 on Day 1

FileName2 = "Pilot1_Day1.xls"
NameOfProcessorFile = "ProcessorFile for all pilots.xls"
Director = "C:\Capt Noel Thesis\1b\"

NumberOfFiles = 22
Windows(NameOfProcessorFile).Activate
Sheets("Main Sheet").Activate
For m = 1 To 22
' Modify the column number below for the appropriate pilot and day combination
  Bookname1(m) = Cells(19 + m, 13).Value
Next m


'******* NO MODIFICATIONS BELOW THIS POINT ********************
'************************************************************

' Determine if the data file is large (approx. 180MB) or if there are 22 smaller files

' *** No need to build this logic in at this point of time *******

    MainCounter = Cells(14, 15).Value
    Filenumber = Cells(15, 15).Value
    Bookname = Bookname1(Filenumber)
    File = Director & Bookname

    ' Loads file here '
    Workbooks.OpenText FileName:=File, Origin:=xlWindows, StartRow:=1, DataType:=xlDelimited,
TextQualifier:=xlDoubleQuote, ConsecutiveDelimiter:=True, Tab:=True, Semicolon:=False,
Comma:=False, Space:=True, Other:=False, FieldInfo:=Array(Array(1, 1), Array(2, 1), Array(3, 1))

' Determine where the data starts in the data files
  i = 1
  While (Cells(i, 26).Value = "")
      i = i + 1
  Wend
  DataStart = i 'This is the first row of the data, but it has labels in this row

' Determine where the VEOG and HEOG columns lie... they are not to be included as processed data
  For i = 1 To 31
    If Cells(DataStart, i).Value = "VEOG" Then
      VEOGLocation = i
    End If

    If Cells(DataStart, i).Value = "HEOG" Then
```

```
      HEOGLocation = i
    End If
  Next i

'Copy the data and close the data file without making any changes to it.
    Range(Cells(1, 1), Cells(33000, 31)).Select
    Selection.Copy
    Windows(NameOfProcessorFile).Activate
    Sheets("Data Sheet").Activate
    Range("Ad1").Select
    ActiveSheet.Paste
    ' This copy is added only to eliminate a message appearing asking whether or not to keep the copied
                                                        info in the clipboard.
    Range("Ad1").Select
    Selection.Copy
    Windows(Bookname).Activate
    ActiveWorkbook.Close

' Delete these two unnecessary data columns
  Windows(NameOfProcessorFile).Activate
  Sheets("Data Sheet").Activate
  Set r1 = Columns(VEOGLocation + 29)
  Set r2 = Columns(HEOGLocation + 29)
  Set myMultiAreaRange = Union(r1, r2)
  myMultiAreaRange.Select
  Selection.Delete Shift:=xlToLeft

' Build the correct order of columns to create a consistent output file with the same EEG order or nodes
' This is the correct order for the 29 EEG nodes (no particular reason for this order, but it will be made the
"correct" order:
' T8  O2  P10 PZ  FP1 C3  PO3 O1  IZ  P4  F3  T7  OZ  FP2 F8  P9  P3  P8  C5  CZ  FZ  F4  C6  F7  P7
FC2 PO4 FC1 C4

  For i = (1) To (29)
    ColumnLabels(i) = Cells(DataStart, i + 29).Value  'start at column 30
  Next i


' Now determine which data column goes where... in order, of course!
' This is just a copy of the list/array from the Build_File macro
  ColumnName(0) = "T8"
  ColumnName(1) = "O2"
  ColumnName(2) = "P10"
  ColumnName(3) = "PZ"
  ColumnName(4) = "FP1"
  ColumnName(5) = "C3"
  ColumnName(6) = "PO3"
  ColumnName(7) = "O1"
  ColumnName(8) = "IZ"
  ColumnName(9) = "P4"
  ColumnName(10) = "F3"
  ColumnName(11) = "T7"
  ColumnName(12) = "OZ"
  ColumnName(13) = "FP2"
  ColumnName(14) = "F8"
  ColumnName(15) = "P9"
  ColumnName(16) = "P3"
  ColumnName(17) = "P8"
  ColumnName(18) = "C5"
  ColumnName(19) = "CZ"
  ColumnName(20) = "FZ"
  ColumnName(21) = "F4"
  ColumnName(22) = "C6"
  ColumnName(23) = "F7"
  ColumnName(24) = "P7"
  ColumnName(25) = "FC2"
  ColumnName(26) = "PO4"
  ColumnName(27) = "FC1"
  ColumnName(28) = "C4"

' Now copy and paste the columns into their correct order
  Matches = 0
  While Matches < 29
    For i = 0 To 28
      For j = 1 To 29
        If ColumnLabels(j) = ColumnName(i) Then
          'Copy the column and put it in its proper location
          'The column to select is j
          'The column to put it in is: i + 1
          Range(Cells(1, j + 29), Cells(32000, j + 29)).Select
          Selection.Copy
          Range(Cells(1, i + 1), Cells(32000, i + 1)).Select
          ActiveSheet.Paste
          Matches = Matches + 1

        End If
      Next j
    Next i
  Wend

' Error check for blank columns here.  If blank set them to 50, which will eliminate an error (type
mismatch) later in processing
  For n = 1 To 29
    CellCheck = Cells(DataStart + 1, n).Value
    If CellCheck = 0 Then
      For o = 1 To 32000
        Cells(DataStart + o, n).Value = 50
      Next o
    End If
  Next n

' Now delete (clear) the rest of the data that we no longer need... hopefully speeds up processing
    Range(Cells(1, 30), Cells(32000, 60)).Select
    Selection.ClearContents

' Grab the appropriate cells from the data file
  DataStart = DataStart + 1
  DataEnd = DataStart + 255
  DataStart2 = DataStart
  DataEnd2 = DataEnd
```

```
For i = 1 To 29        'This is the counter for the number of columns to process in each file
  DataStart = DataStart2
  DataEnd = DataEnd2
  For j = 1 To 120       'Each column has 120 seconds in it
    'Select the appropriate cells
    Range(Cells(DataStart, i), Cells(DataEnd, i)).Select
    Selection.Cut

    ' Put the values into the FFT processor worksheet in the processor file
    Sheets("FFT").Activate
    Range("A2").Select
    ActiveSheet.Paste

    ' Add appostrophies to the any negative values
    For RowNumber = 2 To 257
      CellValue = Cells(RowNumber, 1).Value
      If (CellValue < 0) Then
        CellValue2 = "'" & CellValue
      Else
        CellValue2 = CellValue
      End If

      Cells(RowNumber, 2).Value = CellValue2

    Next RowNumber
    ' Clear the old FFT data: eliminates the alert message that would otherwise appear when doing
                                                   another FFT on top of it
    Range(Cells(2, 3), Cells(257, 3)).Select
    Selection.ClearContents

    ' Perform FFT on the data
    Application.Run "ATPVBAEN.XLA!Fourier", ActiveSheet.Range("$B$2:$B$257"), _
      ActiveSheet.Range("$C$2"), False, False

    ' Stick the results into the respective arrays
    DeltaArray(j) = Cells(261, 8).Value
    ThetaArray(j) = Cells(261, 9).Value
    AlphaArray(j) = Cells(261, 10).Value
    BetaArray(j) = Cells(261, 11).Value
    UBetaArray(j) = Cells(261, 12).Value

    ' Get the variables ready for the next j iteration, and re-activate the data file
    DataStart = DataStart + 256
    DataEnd = DataEnd + 256
    Sheets("Data Sheet").Activate

  Next j

  'Calculate the 10 second averages.
   Sheets("FFT").Activate

  '***** Calculate the odd exemplars first *****

  d = 0  'This counter keeps track of cell location in the processor file
```

```
For b = 0 To 11 'This is the number of odd exemplars per 2 minute interval (12)
  For c = 1 To 10 'this is the number of seconds per time window
    AverageD = AverageD + DeltaArray(c + d)
    AverageT = AverageT + ThetaArray(c + d)
    AverageA = AverageA + AlphaArray(c + d)
    AverageB = AverageB + BetaArray(c + d)
    AverageU = AverageU + UBetaArray(c + d)
  Next c
  'Put the averages into the correct cells in the processor file worksheet
  Cells(265, 2 + 2 * b).Value = AverageD / 10
  Cells(266, 2 + 2 * b).Value = AverageT / 10
  Cells(267, 2 + 2 * b).Value = AverageA / 10
  Cells(268, 2 + 2 * b).Value = AverageB / 10
  Cells(269, 2 + 2 * b).Value = AverageU / 10
  d = d + 10  'Increments where in the arrays to find the right data
  AverageD = 0    'These are reset for every b value; we want fresh 10 second interval values
  AverageT = 0
  AverageA = 0
  AverageB = 0
  AverageU = 0
Next b

'***** Calculate the even exemplars next *****

d = 4  'The even windows start at 5 seconds (so when c=1, c + Maincounter2 = 5 seconds)
For b = 0 To 10 'This is the number of even exemplars per 2 minute interval (11 of them)
  For c = 1 To 10 'this is the number of seconds per time window
    AverageD = AverageD + DeltaArray(c + d)
    AverageT = AverageT + ThetaArray(c + d)
    AverageA = AverageA + AlphaArray(c + d)
    AverageB = AverageB + BetaArray(c + d)
    AverageU = AverageU + UBetaArray(c + d)
  Next c
  'Put the averages into the correct cells in the worksheet
  Cells(265, 3 + 2 * b).Value = AverageD / 10
  Cells(266, 3 + 2 * b).Value = AverageT / 10
  Cells(267, 3 + 2 * b).Value = AverageA / 10
  Cells(268, 3 + 2 * b).Value = AverageB / 10
  Cells(269, 3 + 2 * b).Value = AverageU / 10
  d = d + 10  'Increments where in the arrays to find the right data
  AverageD = 0    'These are reset for every b value; we want fresh 10 second interval values
  AverageT = 0
  AverageA = 0
  AverageB = 0
  AverageU = 0
Next b

' Grab the log10 of these average values from the processor sheet and put them into the processed data
                                                   file
'First put the processed data into arrays
For b = 0 To 22
  DeltaArray(b + 1) = Cells(271, 2 + b).Value
  ThetaArray(b + 1) = Cells(272, 2 + b).Value
  AlphaArray(b + 1) = Cells(273, 2 + b).Value
  BetaArray(b + 1) = Cells(274, 2 + b).Value
```

```
      UBetaArray(b + 1) = Cells(275, 2 + b).Value
    Next b

    'Now put these arrays into the correct places in the processed data file

    Windows(FileName2).Activate
    For b = 1 To 23
      Cells(2 + (b - 1) + MainCounter * 23, 11 + (i - 1) * 5).Value = DeltaArray(b)
      Cells(2 + (b - 1) + MainCounter * 23, 12 + (i - 1) * 5).Value = ThetaArray(b)
      Cells(2 + (b - 1) + MainCounter * 23, 13 + (i - 1) * 5).Value = AlphaArray(b)
      Cells(2 + (b - 1) + MainCounter * 23, 14 + (i - 1) * 5).Value = BetaArray(b)
      Cells(2 + (b - 1) + MainCounter * 23, 15 + (i - 1) * 5).Value = UBetaArray(b)
    Next b

    Windows(NameOfProcessorFile).Activate
     Sheets("Data Sheet").Activate
  Next i  'This loops for the next column of data within the data file

'Save the processed data file (just in case!)
    Windows(FileName2).Activate
    ActiveWorkbook.Save

    'Update the main counter value and filenumber in the main sheet before closing down excel
    Windows(NameOfProcessorFile).Activate
    Sheets("Main Sheet").Activate
    Cells(14, 15).Value = MainCounter + 1
  Cells(15, 15).Value = Filenumber + 1
  Application.ScreenUpdating = True

  ' Delete the data on the data sheet of the processor file so that it isn't so large

  Sheets("Data Sheet").Activate
  Range(Cells(1, 1), Cells(32000, 29)).Select
  Selection.ClearContents
  Sheets("Main Sheet").Activate
   ActiveWorkbook.Save

End Sub
```

Appendix B. Additional Information For Working With SNNAP

B.1. Getting the Weights Out From SNNAP

1. Go to the Network menu of SNNAP

2. Click on "Text Save" and give a file name for the weights to be placed into

3. Go to program like Microsoft Excel and open the file as "Space Delimited"

4. The first two rows show the structure of the ANN. The first row is the number of layers in the network. The second row shows: the number of input nodes, the number of middle nodes, and the number of output nodes

5. Select and delete rows 1 through 14, keeping row 15. (Note: rows 12 through 14 might mean something to the ANN, but they are not the weights going into or out of the hidden nodes.)

6. The next set of rows and columns are the hidden node weights. The information below will help organize them, as well as explain their location. First make sure that all of the rows start in the same column. Usually this means "cutting and pasting" several rows so that they all start in column 2. Each column represents one input node, listed in the order identified from the data set. (Example: "From Input Variable X"). Each row represents one hidden node (Example: "To hidden Node Y"). The last column of the weights should be deleted. (Note: despite numerous attempts to confirm that this column is related to a bias term included in the model structure, the attempts have failed. It is possible that somehow it is related.)

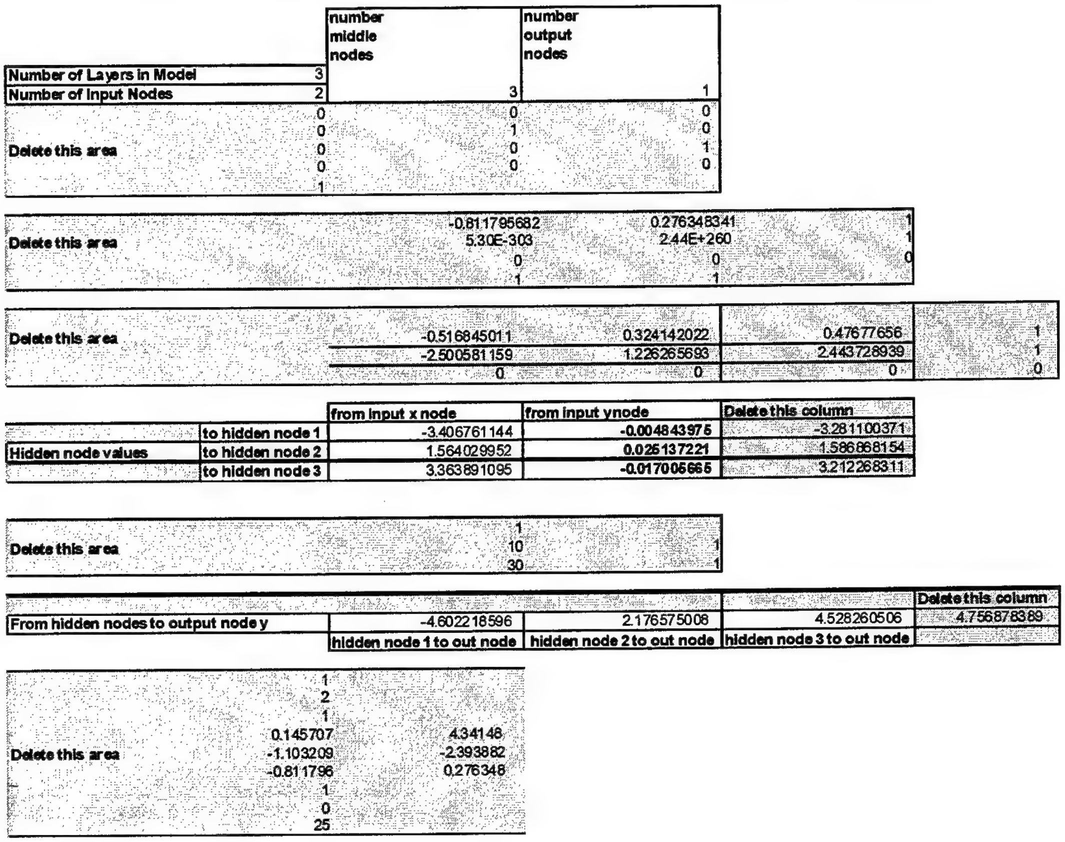
7. At the end of the hidden node weights, look for a "1" all by itself in either column 1 or column 2. Select this row and the next three rows. Delete all four rows.

8. The next few rows in the spreadsheet are the output weights, with the number of rows varying depending on the number of outputs in the network structure. If there are more rows than outputs in the network, the first row of this group should be ignored.

9. All of the remaining rows in the file can be deleted.

10. The example output file shown in Table B-1 highlights the key items mentioned in the steps above. All of the gray areas should be deleted. Extra spaces have been added to show the different areas of the output file.

Table B-1. Example of Identifying Middle Layer Weights in SNNAP



		number middle nodes	number output nodes		
Number of Layers in Model	3				
Number of Input Nodes	2	3	1		
	0	0	0		
	0	1	0		
Delete this area	0	0	1		
	0	0	0		
	1				

		-0.811795682	0.276348341	1	
Delete this area		5.30E-303	2.44E+260	1	
		0	0	0	
		1	1		

Delete this area		-0.516845011	0.324142022	0.47677656	1
		-2.500581159	1.226265693	2.443728939	1
		0	0	0	0

		from input x node	from input ynode	Delete this column
	to hidden node 1	-3.406761144	-0.004843975	-3.281100371
Hidden node values	to hidden node 2	1.564029952	0.025137221	1.586868154
	to hidden node 3	3.363891095	-0.017005665	3.212268311

		1	
Delete this area		10	1
		30	1

					Delete this column
From hidden nodes to output node y		-4.602218596	2.176575008	4.528260506	4.756878389
		hidden node 1 to out node	hidden node 2 to out node	hidden node 3 to out node	

	1	
	2	
	1	
	0.145707	4.34148
Delete this area	-1.103209	-2.393882
	-0.811796	0.276348
	1	
	0	
	25	

B.2. Building the Signal-To-Noise Ratios with the Hidden Layer Weights From SNNAP

1. First find and isolate the hidden layer weights in the weights output file using the process from Part A of this appendix.

2. Remember that each column of hidden weights represents the relationship of one input node to all of the hidden nodes.

3. Using the SUMSQ function in Excel, square each value in each column and sum the columns. Make sure that each column is clearly labeled, including the noise input variable column. A good suggestion involves placing column labels above the respective columns.

4. In the next row calculate the incomplete signal-to-noise ratios. This is simply the sum total from each input variable column found in Step #3 above divided by the sum total from the noise column, also found in Step #3 above.

5. In the next row, finish the signal-to-noise ratios by taking the $\log_{10}$ of the above ratio and multiply the result by 10. This row now represents the signal-to-noise ratios.

6. At this point one can sort the signal-to-noise ratios from greatest to least, or vice versa. Regardless of the sort criteria, we now identify those input variables that are more “important” than others by looking at the signal-to-noise ratios, where larger values represent more “important” variables than those with smaller values. (Note: if one sorts the ratios at this point, first “cut” the data and then select “paste special, values”. This enables a sort of the values to occur properly.)

7. An example of this process is shown in Table B-2. Step #8 discusses the example.

8. From these ratios, one can clearly see that the input variable "interblink" is the most significant input variable, with "interbreath" being the least important. The negative value of the SNR from "interbreath" indicates that it provides less information than the "noise" input variable. This means that "interbreath" is of little help for classification and can be dropped from the network with negligible impact.

B.3. Build A Confusion Matrix Using The Projection Command

SNNAP has several “bugs” in it that keep one from consistently using a separate file as a second validation data set. (SNNAP will cause an error that requires it to be shut down.) Possible ways to avoid this error include placing the data files in directories not more than 2 levels away from the c:\ root directory, and not using directory names longer than 8 characters. Should errors continue to occur, it is often fastest to ignore this separate validation data set option. Instead, train the network using only the training and test-training options that are set up as defaults when you build a network with SNNAP, and then run a “projection” with the separate data set after training the network. With

Table B-2. Example of Calculating Signal-to-Noise Ratios

	Original Hidden Node Weights						
	bpm	hrt var	blinks	interblink	breath	interbreath	noise
to hidden node 1	1.228182	-4.66408	0.873662	-0.5890601	-1.98919	-1.91096867	-0.72471
to hidden node 2	0.44289	-0.88165	-1.59677	-2.6500522	1.344144	0.037204876	0.366523
to hidden node 3	6.945779	-7.15632	4.203261	-3.3273452	-1.56593	3.131863355	0.395718
to hidden node 4	-0.13592	-0.25499	-0.07266	-1.0541176	0.172265	-0.68911343	-0.21842
to hidden node 5	-0.242743	-0.18252	0.57865	-0.6447437	0.285281	-0.35842223	0.050174
to hidden node 6	-1.088486	1.741637	-7.30336	-3.9605374	3.9593	-0.49591274	3.528583
to hidden node 7	-8.182255	1.311842	4.565304	-3.0484271	-0.51907	-1.08875646	0.317501
to hidden node 8	0.765886	-0.69689	0.918963	-1.0542315	-1.18348	0.489961823	-0.31332
to hidden node 9	-0.344955	-1.66559	0.293987	0.03797538	-1.38783	-1.20124922	-1.2274
to hidden node 10	2.58347	-2.50951	-4.98877	-2.0563215	3.188603	2.221960279	1.288096
to hidden node 11	-0.537035	-1.61175	0.457642	-0.0166564	-1.54997	-1.13726906	-1.4846
to hidden node 12	1.956048	-6.18525	-5.02061	-9.7290162	-3.90286	0.950555856	0.919123
to hidden node 13	-3.074462	2.910491	2.185458	-4.3110999	2.079584	0.749845554	-2.81503
to hidden node 14	-5.813756	-5.06899	-1.78737	-17.851689	-2.62473	-3.49420204	1.039999
to hidden node 15	0.406276	-0.73725	-1.27231	-2.2929607	1.19505	-0.02816358	0.261513
to hidden node 16	0.714273	-1.36032	-2.60155	-3.2294989	1.826056	0.361938384	0.730456
to hidden node 17	-0.197278	-0.31445	0.158358	-0.433888	-0.07826	-0.8389498	-0.31659
to hidden node 18	-0.372884	-1.4776	0.400758	-0.289906	-1.9721	-0.71053436	-2.08259
to hidden node 19	-1.312679	-2.74164	-6.5734	-14.696181	-1.12337	-3.30825064	-5.85071
to hidden node 20	-5.931768	1.509958	3.888034	-5.5369369	-3.94712	-3.34798413	-5.22659
to hidden node 21	2.452297	1.03936	1.013676	0.77617153	5.735868	0.692736126	-3.45713
to hidden node 22	4.731014	-0.20389	-4.58325	-4.7626373	3.519942	4.523428012	0.654364
to hidden node 23	-2.961659	3.280136	8.136055	-1.6451507	-4.51668	-4.28997996	0.452325
to hidden node 24	-0.086306	-0.24167	0.100493	-0.7876493	0.260308	-0.57004067	-0.13948
to hidden node 25	0.652472	-1.4347	-2.76032	-3.5612125	1.920864	0.349636858	0.794226
to hidden node 26	5.481898	1.598198	-0.33567	0.23240459	3.556493	-1.60769427	0.98622
to hidden node 27	0.110445	-0.43607	-0.61845	-1.8597694	0.72694	-0.35750469	0.043856
to hidden node 28	-0.543087	-1.18985	0.218928	-0.0089165	-0.97949	-1.16975391	-1.30579
to hidden node 29	-0.150824	-0.25705	0.16413	-0.6344681	0.085096	-0.72135071	-0.21748
Sum of Squares Values	**278.6506**	**195.8714**	**318.0347**	**788.015053**	**175.7062**	**104.9728154**	**111.2859**

Input Variables	bpm	hrt var	blinks	interblink	breath	interbreath
Ratios	2.503916	1.760073	2.857816	7.08099395	1.578871	0.943271157
SNR: 10*log(ratio)	**3.986197**	**2.455308**	**4.560342**	**8.50094223**	**1.983467**	**-0.25363445**

this "projection" output, one can then quickly build a confusion matrix to see the desired results.

The construction of the confusion matrix is relatively easy. "If-then" logic statements will need to be added to the output file using a spreadsheet program like Microsoft Excel. These statements will need to split the results into 4 columns based on prediction versus actual values. Section 4-2 addresses the confusion matrix and includes an equation that will need to be placed at the bottom of the columns, after they are summed, to complete the development of the confusion matrix.

Appendix C. Factor Loadings for Factor 2 Physiological Features

The letter “A” indicates the results when using the data set for Pilot 1 on day 1; “B” indicates Pilot 1 on day 2; “X” indicates Pilot 4 on day 1; “Y” indicates Pilot 4 on day 2; “1” indicates Pilot 1 over both days of data; and “4” indicates Pilot 4 over both days of data.

Table C-1. Factor Loadings for Factor 2 Physiological Features

Feature	Data Set	Loading
Blinks	A	0.475
	1	0.492
BPM	B	0.435
Breaths	Y	-0.483
Heart Variability	B	-0.430
Interblink	A	-0.492
	1	-0.476
Interbreath	B	0.487
	Y	-0.404

Appendix D. Ocular and Cardiac Feature Graphs For Pilots 1 and 4 on Days 1 and 2

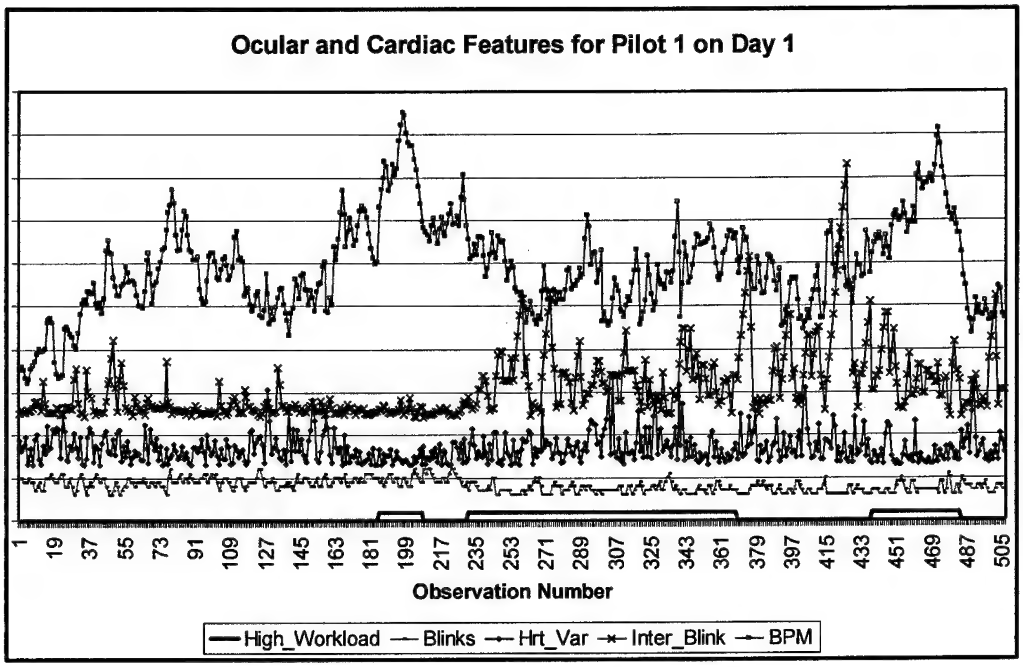


Figure D-1. Ocular and Cardiac Features for Pilot 1 on Day 1

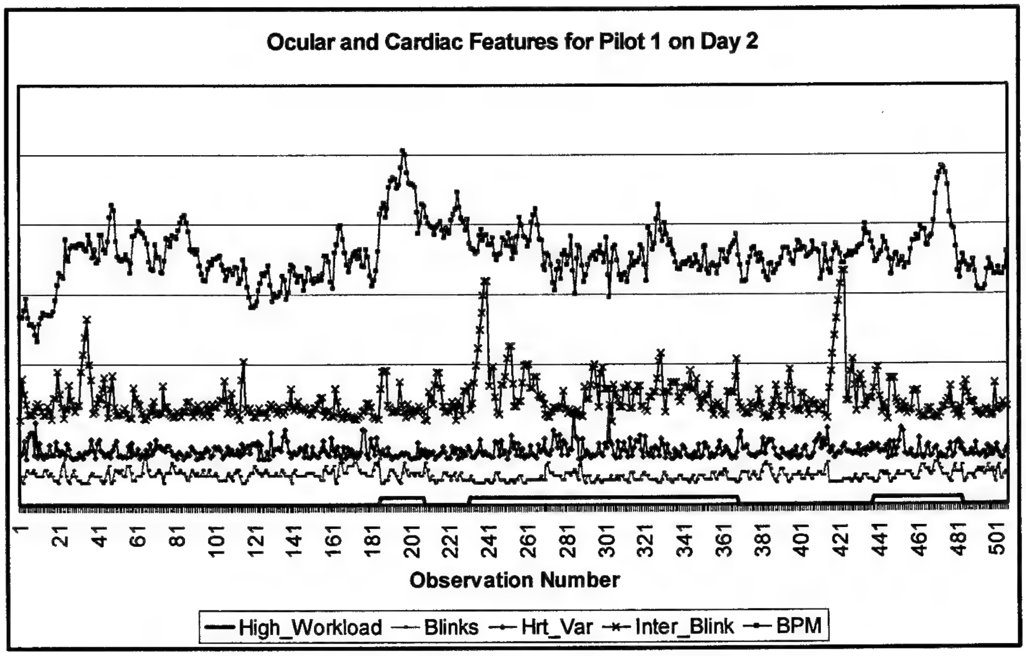


Figure D-2. Ocular and Cardiac Features for Pilot 1 on Day 2

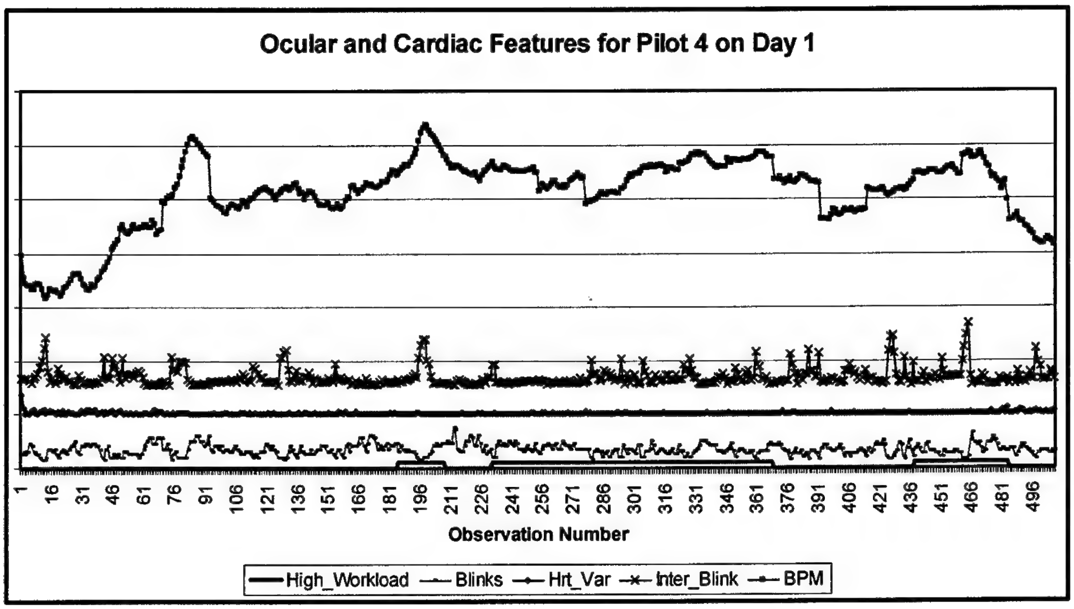


Figure D-3. Ocular and Cardiac Features for Pilot 4 on Day 1

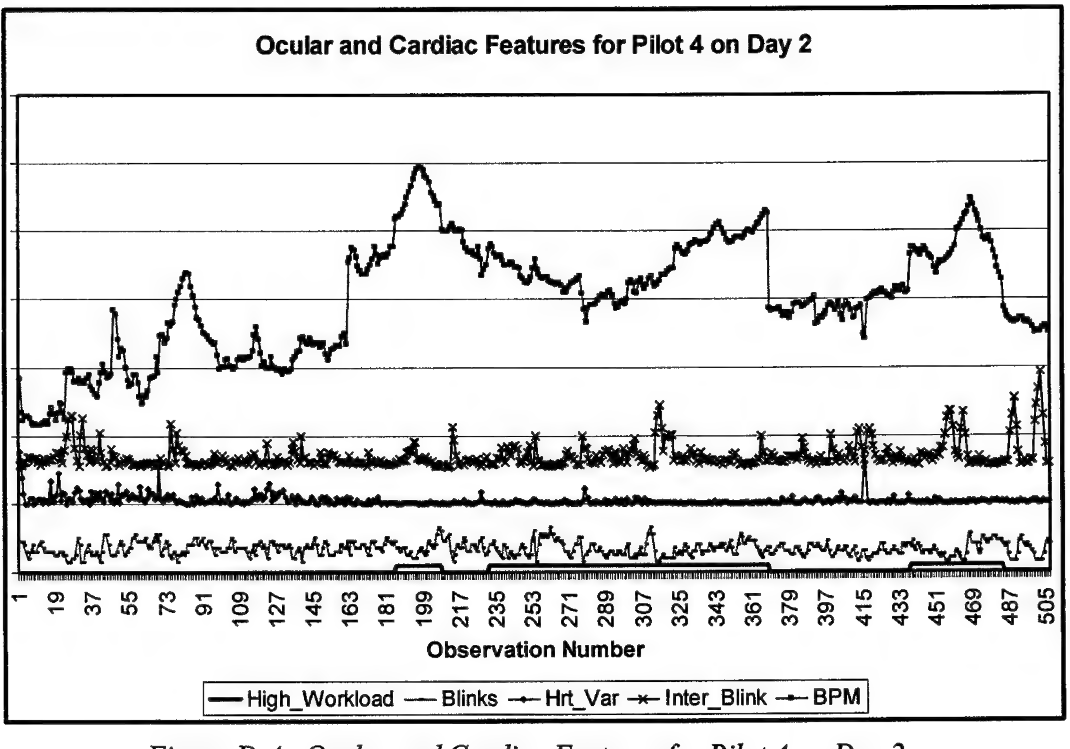


Figure D-4. Ocular and Cardiac Features for Pilot 4 on Day 2

Appendix E. Individual Calibration Scheme To Baseline Comparisons

The information tables for Table E-1 and E-2 are shown in Tables 5-2 and 5-7, respectively.

Table E-1. Baseline CA and Variance Results By Data Set

		Projection Data Set			
		Pilot 1, Day 1	**Pilot 1, Day 2**	**Pilot 4, Day 1**	**Pilot 4, Day 2**
Training Data Set	**Pilot 1, Day 1**		CA = 63.24%	CA = 47.43%	CA = 66.80%
	Pilot 1, Day 2	CA = 64.43%		CA = 48.22%	CA = 72.92%
	Pilot 4, Day 1	CA = 59.09%	CA = 59.09%		CA = 60.87%
	Pilot 4, Day 2	CA = 60.87%	CA = 61.86%	CA = 53.16%	

Average CA Value	**59.83%**
CA Variance	**53.92**

Table E-2. Calibration Scheme CA and Variance Results By Data Set

		Projection Data Set			
		Pilot 1, Day 1	**Pilot 1, Day 2**	**Pilot 4, Day 1**	**Pilot 4, Day 2**
Training Data Set	**Pilot 1, Day 1**		CA = 69.76%	CA = 73.72%	CA = 71.34%
	Pilot 1, Day 2	CA = 69.57%		CA = 68.77%	CA = 69.57%
	Pilot 4, Day 1	CA = 71.54%	CA = 71.15%		CA = 72.33%
	Pilot 4, Day 2	CA = 75.69%	CA = 75.69%	CA = 75.10%	

Average CA Value	**72.02%**
CA Variance	**6.23**

Bibliography

1. Air Force Research Laboratory, AFRL, "Flight Psychophysiological Laboratory," 1998. Office Brochure, Flight Psychophysiological Laboratory, Human Interface Technology Branch, Crew System Interface Division, Human Effectiveness Directorate (AFRL/HECP).

2. Auten, J. "G-LOC: Is the Cluebag Half Full or Half Empty?," *Flying Safety*, 52:5-6 (1996).

3. Basar, E., et al. "Sensory and Cognitive Components of Brain Resonance Responses," *Acta Otolaryngol*, 491:25-35 (1991).

4. Bauer, K.W., "OPER 685, Applied Multivariate Data Analysis," Fall 2000. Air Force Institute of Technology, Wright-Patterson AFB OH.

5. Bauer, K.W., et al. "Feature Screening using Signal-to-Noise Ratios," *Neurocomputing*, 31:29-44 (1999).

6. Belue, L.M. and K.W. Bauer. "Determining Input Features for Multilayer Perceptrons," *Neurocomputing*, 7:111-121 (1995).

7. Bishop, Christopher M. *Neural Networks for Pattern Recognition.* Oxford UK: Clarendon Press, 1998. 83-84.

8. Burden, R.L. and J.D. Faires. *Numerical Analysis* (Sixth Edition). Boston MA: PWS Publishing Co., 1997.

9. Demuth, H., Beale, M. *Neural Network Toolbox version 3 User's Guide*, The Math Works Inc., 1998.

10. East, J.A. *Feature Selection for Predicting Pilot Mental Workload.* MS thesis, School of Engineering, March 2000.

11. Greene, K.A., Bauer, K.W., Wilson, G.F., Russell, C.A., Rogers, S.K., and Kabrisky, M., "Selection of psychophysiological features for classifying air traffic controller workload in neural networks," *International Journal of Smart Engineering System Design*, to appear (1998).

12. Greene, K.A. *A Feature Saliency in Artificial Neural Networks with Application to Modeling Workload.* PhD dissertation, Air Force Institute of Technology, Wright-Patterson AFB OH, December 1998.

13. Halberg, B., et al. *Using Microsoft Excel 97: Special Edition.* Indianapolis IN: Macmillan Computer Publishing, 1997.

14. Hankins, T.C. and Wilson, G.F. “A Comparison of Heart Rate, Eye Activity, EEG, and Subjective Measures of Pilot Mental Workload during Flight,” *Aviation, Space, and Environmental Medicine*, 69:360-367 (1998).

15. Laine, T.I. *Selection of Psychophysiological Features Across Subjects for Classifying Workload Using Artificial Neural Networks.* MS thesis, School of Engineering, March 1999.

16. Laine, T.I., Bauer, K.W., and Lanning, J.W., 1999, “Multiple Crewmember Workload Classification using Neural Networks with Input Feature Selection,” *Intelligent Engineering Systems Through Artificial Neural Networks*, Proceedings of Artificial Neural Networks in Engineering (ANNIE) International Conference, St. Louis MO, 7-10 Nov 1999.

17. Math Works, Inc. *MATLAB Signal Processing Toolbox User's Guide.* Natick MA: Math Works, 1998. 2-2, 3-5—3-8.

18. Sall, J., Lehman, A., *JMP Start Statistics: A Guide to Statistics and Data Analysis Using JMP Software.* Belmont CA: Druxbury Press, 1996.

19. Smith, M. *Neural Networks for Statistical Modeling.* Boston MA: International Thomson Computer Press, 1996.

20. Steppe, J.M. *Feature and Model Selection in Feedforward Neural Networks.* PhD dissertation, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, 1994.

21. Steppe, J.M. and Bauer, K.W., “Improved Feature Screening in Feedforward Neural Networks,” *Neurocomputing*, 13:47-58 (1996).

22. Swingler, K. *Applying Neural Networks: A Practical Guide.* San Diego CA: Academic Press, 1996.

23. The American Heritage College Dictionary, Second Edition, “dendrite.” Houghton Mifflin Company, Boston MA, 1985.

24. The American Heritage College Dictionary, Second Edition, “physiological.” Houghton Mifflin Company, Boston MA, 1985.

25. Thaller, R.A., Petsche, H., Rappelsberberger, P., Pockberger, H., Lindner, K., and Imhof, H., "An Approach to a Synopsis of EEG Parameters, Morphology of Brain Convolutions and Mental Activities," *Brain Topography*, 4(1):65-73 (1991).

26. Wackerly, D.D., et al. *Mathematical Statistics with Applications*. Belmont CA: Wadsworth Publishing Co., 1996.

27. Wiggins, V.L., Borden, K.M., Turner, K.L., Looper, L.T., and Grobman, J.H. *Statistical Neural Network Analysis Package (SNNAP) Version 2.0 User's Manual*, Interim Technical Paper- July 1994 – July 1995, Air Force Material Command, Brooks Air Force Base TX, 1996.

28. Wilson, G.F. "Applied Use of Cardiac and Respiration Measures: Practical Considerations and Precautions," *Biological Psychology*, 34:163-178 (1992).

29. Wilson, G.F. "Air-to-Ground Training Missions: A Psychophysiological Workload Analysis," *Ergonomics*, 36(9):1071-1087 (1993).

30. Wilson, G.F., Fisher, F., "Cognitive Task Classification Based Upon Topographical EEG Data," *Biological Psychology*, 40:239-250 (1995).

31. Wilson, G.F., Fullenkamp, P. and Davis, I. "Evoked Potential, Cardiac, Blink, and Respiration Measures of Pilot Workload in Air-to-Ground Missions," *Aviation, Space and Environmental Medicine*, February, 100-105 (1995).

32. Wilson, G.F., Gundel, A. "Topographical Changes in the Ongoing EEG Related to the Difficulty of Mental Tasks," *Brain Topography*, 5:17-25 (1992).

33. Zill, Dennis G., *A First Course in Differential Equations with Modeling Applications* (Sixth Edition). Pacific Grove CA: Brooks/Cole Publishing Co., 1997.

Vita

Captain Jeremy Noel was born and raised in Northern California. In 1990, he graduated from Monta Vista High School in Cupertino, California, before attending the United States Air Force Academy, class of 1994. Following graduation in 1994 with a Bachelor of Science degree in Operations Research, he was assigned to the Directorate of Requirements division of Headquarters Air Combat Command at Langley Air Force Base, Virginia, as a Joint Command and Control Systems Analyst. After four years of analyzing the architectures, integration, and fielding of command, control, communications, computers, and intelligence (C4I) systems for the Combat Air Forces, he received a new assignment with the Air Force Command and Control Training and Innovation Center (AFC2TIC) at Hurlburt Field, Florida. As the Deputy Technical Advisor for test and evaluation supporting both AFC2TIC and the Aerospace Command and Control & Intelligence, Surveillance and Reconnaissance Center (AC2ISRC), he focused on improving test methods and analytical techniques to solve critical command and control issues. In August 1999, Jeremy entered the Air Force Institute of Technology, and following graduation he will be assigned as an analyst in the Modeling, Simulation, and Analysis Branch of the Developmental Planning Directorate of Space and Missile Systems Center, Los Angeles Air Force Base, California. Jeremy's hobbies include numerous sports and activities such as swimming, waterpolo, sailing, skiing, and stock investing.

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 074-0188*

1. REPORT DATE *(DD-MM-YYYY)* 20-03-2001	**2. REPORT TYPE** **Master's Thesis**	**3. DATES COVERED** *(From – To)* Jun 2000 – Mar 2001

4. TITLE AND SUBTITLE PILOT MENTAL WORKLOAD CALIBRATION	**5a. CONTRACT NUMBER** **5b. GRANT NUMBER** **5c. PROGRAM ELEMENT NUMBER**
6. AUTHOR(S) Noel, Jeremy B, Capt, USAF	**5d. PROJECT NUMBER** **5e. TASK NUMBER** **5f. WORK UNIT NUMBER**
7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S) Air Force Institute of Technology Graduate School of Engineering and Management (AFIT/EN) 2950 P Street, Building 640 WPAFB OH 45433-7765	**8. PERFORMING ORGANIZATION REPORT NUMBER** **AFIT/GOR/ENS/01M-12**
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Glenn F. Wilson, PhD. AFRL/HECP 2255 H. Street, Bldg 33 Wright-Patterson AFB, OH 45433-7022 (937) 785-8748 Glenn.Wilson@wpafb.af.mil	**10. SPONSOR/MONITOR'S ACRONYM(S)** **11. SPONSOR/MONITOR'S REPORT NUMBER(S)**

12. DISTRIBUTION/AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Predicting high pilot mental workload is important to the U.S. Air Force because lives and aircraft can be lost when errors are made during periods of mental overload and task saturation. Current research efforts use psychophysiological measures such as electroencephalography, cardiac, ocular, and respiration measures in an attempt to identify and predict mental workload levels.

The primary focus of this effort is the development of a calibration scheme that allows a small subset of salient psychophysiological features developed using actual flight data for one pilot on a given day to accurately classify pilot mental workload for a separate pilot on a different day. To accomplish this objective, the signal-to-noise ratio feature screening method is employed to determine the usefulness of 151 psychophysiological features in feed-forward artificial neural networks. Factor analysis identifies patterns in features that vary with changes in workload level. Methodologies for workload level modification and data calibration are presented and tested. Our results indicate the calibration scheme can increase classification accuracy (CA) over 55%, decrease CA variance by 88%, and decrease by 88% the number of features to process than previous classification methods and classifiers.

15. SUBJECT TERMS
workload; measurement (mental); mental stress; physiology; psychophysiology; neural nets; calibration

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** **Bauer, K.W., Dr AFIT/ENS**
a. REPORT U	**b. ABSTRACT** U	**c. THIS PAGE** U	UU	180	**19b. TELEPHONE NUMBER** *(Include area code)* (937) 255-6565, ext 4328 Kenneth.Bauer@afit.af.edu